SAINT LOUIS UNIVERSITY

AUTOMATED PILL SORTING

May 15, 2022

ROBERT MCCORMICK
TIMOTHY METZGER
CHASE NIECE
AKASH SOOKUN

DEPARTMENT OF AEROSPACE AND MECHANICAL ENGINEERING
PARKS COLLEGE OF ENGINEERING, AVIATION AND TECHNOLOGY
SAINT LOUIS UNIVERSITY

Executive Summary

In the following report on improving pharmaceutical drug adherence through pill sorting and dispensing devices a need analysis is conducted for the project, multiple conceptual designs are proposed along with a final recommendation, and finally an embodiment design is given. Included within the need analysis is the need statement, target customer/product, function structure, constraints, order of magnitude estimations, and requirements.

In summary, the need statement of this project addresses the continual problem faced by the ill who struggle to adhere to their prescribed medication routine due to failure to obtain medication from their pharmacy and/or failure to manage their medications. The target customer for this project is pharmacies which will provide them with an economical pill sorting machine that will take patients pills and package them into a medication dispensing compatible parcel. As for the function structure, there are four main functions, grabbing pills, confirming pills, distributing pills, and counting dosage. Each of these functions have two sub-functions and more beyond that. For a graphical representation of this structure see the function structure section located within the need analysis section. As for the constraints there are many that concern the wide range of pill shapes and sizes. Additionally, some of the biggest constraints that this project will see are due to the regulations that are imposed by the FDA and DEA. In the order of magnitude estimations, two configurations (a claw mechanism and a suction tip) for the pill sorting head are investigated to determine their feasibility. From the calculations it is found that both of the proposed configurations are valid and pose no risk to harming the medication. Additionally in this section calculations are performed to determine a rough estimate of operating cost for the machine and compared to the cost associated with pharmacy technicians. Finally, the requirements section addresses the key metrics needed for the success of the project. These numbers can be found in the requirement section.

For the conceptual design section of this paper three distinct solutions are proposed to meet the demands of the problem. The common elements among the three designs is the use of a surface mount technology (SMT) pick and place head to move medications from one location to another, and the sorted pill bin being a circular disk divided into time frames such as the days of the week. In the first concept pills are fed into individual channels via a hopper aligning the pills single file. At the bottom of the channels an SMT head moves to the channel location, grabs the pill, and returns that pill to its sorted location. For concept #2 an SMT head is placed in the center of a circular arc where pill bins are placed along that arc. To retrieve the medications the SMT head rotates through the use of servo at its base to the appropriate pill location, retrieves the pill, and returns it to the sorted location. Finally, for concept #3 the SMT head is placed onto a gantry above pill bins which are designed so that a standardized pharmaceutical medication bottle may be directly inserted for sorting. To sort the pills the gantry positions itself over the appropriate location, grabs the pill, and finally drops the pill onto a conveyor that will deposit the pill into its appropriate sorted location. For visual representations of these three concepts please see the conceptual design section.

In recommending one of these designs the team had notable issues with both concepts #1 and #3. The primary concern with both of these designs is that in the event that medication is not properly managed it could end up in the improper location leading to potential harm to the end consumer. In the first design since the pills are positioned adjacent to each other there is potential for mixing both during the pill feed and transport. As for concept #3 the conveyor belt system is concerning as medication could easily move while on the belt. This gravitated the team toward concept #2, but to verify our suspicion a Pugh matrix was used. In doing so the teams affinity toward concept #2 was verified and thus it is the concept recommended for further development.

Next, in the embodiment design section of this report the recommended design for further development from the conceptual design section is improved upon. In this section the estimated size and type of the electrical components is fleshed out leading to a final 3D model. Additionally, in this model modifications are made to the main sorting arm that are greatly inspired by SCARA robotic arms. In this section can also be found detailed analysis of failure modes, assembly details, stress analysis on the main sorting arm, software, test results, and final prototype cost. From these sections there are many key takeaways. Firstly, through stress analysis conducted by applying a transverse loading at the sorting arms tip it is found that the main sorting arm when constructed of PLA will be capable of withstanding ~20kg of transverse loading at the far end. Second, in the software section, a user interface is created using PyQT to streamline the user's interaction with the machine and ensure the proper loading of medication. Next, in the test plan and verification section, the machine was tested in 4 phases with differing amounts of pills and pill types. In testing, the machine was found to perform well with tablets but not capsules achieving sorting accuracies of ~70% when sorting tablets and ~25% when sorting multiple medication types. This result was below expectation and thus requires further development before being a viable product for pharmacies. Lastly, the final prototype cost is detailed part by part coming to a total below $500.

Lastly, there is the final product section which details areas for further improvement in design as well as packaging, manufacturing details, and cost estimates. Some notable improvements to be made would be to include further safety measures to protect the user and the inclusion of computer vision to aid in pill retrieval cycles. As for packaging, the desired method of delivering the product would be in pieces with the major more complex components, such as the central sorting head, pre-assembled. This method of delivery would help keep the complex components safe from damage during transport and given the current design would be easily assembled. For manufacturing of the product there are only two major processes that need to be completed. The first is to CNC cut the base plate to mount all the other components to. The second is to create the plastic components using polypropylene and injection molding. Lastly, a final cost estimate is provided for the machine at ~250$.

Table of Contents

Introduction **6**

Need Analysis **6**

- Need Statement: 6
- Target Customer and Product: 7
- Function Structure: 7
- Constraints: 10
- Order of Magnitude Estimations: 11
- Requirements: 15

Conceptual Design **16**

- Introduction 16
- Concept #1 16
- Concept #2 22
- Concept #3 35
- Concept Evaluation & Recommendation 39
- Further Development 41

Embodiment Design **48**

- Final Concept Description 48
- Sizing 61
- Failure Mode Effects Analysis 66
- Assembly & Part Details 67
- Analysis 75
- Software 84
- Test Plan & Verification 92
- Prototype Cost 95

Final Product **96**

- Recommended Final Product Information 96
- Packaging 97
- Manufacturing & Assembly Instructions 99
- Cost Estimates 99

ABET Outcomes **101**

Appendix 1 - Order of Magnitude Estimations **104**

Appendix 2 - Function Chart **105**

Appendix 3 - Conceptual Design Calculations **106**

Appendix 4 - Embodiment Design Calculations **107**

Appendix 5 - Software Code **108**

Appendix 6 - Patents **133**

Appendix 7 - Ideas Generated **146**

Appendix 8 - Rendered Images **154**

I. Introduction

For so many people in the U.S. and across the world, the process of taking daily medication is a burden that presents unexpected challenges. While at face value this task seems simple, the reality is that people struggle to make this experience easy. People shouldn't be consumed by their need to take medication, nor should their thoughts revolve around the difficulty of the process. Solving issues people experience due to the distribution of their medication reduces stress on all of those who are involved, and ultimately provides independence for the end user. Finding a solution that is implemented at the pharmacy level would help people be able to collect all of their medications and use them at home without any effort in distributing or sorting the pills.

II. Need Analysis

Need Statement:

A rampant problem not tied to any specific demographic is that of pharmaceutical adherence. For many simply remembering to pick up their prescriptions can be difficult while for others it is sorting through their many medications and keeping things in order. It is not an uncommon occurrence for those taking many medications to forget if they have taken their medication and to skip a dose or take another. This problem has long had the simple solution of the patient picking up their prescriptions in individually packaged vials and going home and sorting them all into bins using devices such as the one pictured below.

Figure 1: Pill Organizer [1]

Automated pill sorting is not a new idea and has been used in larger scale environments such as hospitals and hospice care. As mentioned earlier a contributing factor to inhibiting medication adherence is getting people to simply pick up their medication and for many this has to be done multiple times a month increasing difficulty. Due to the struggle people have with picking up their medication and then sorting it on their own, it leads to the need for a streamlined process. This process ideally would look like all of the patients prescriptions being synced for pick up at the same time and coming pre-sorted in a manner similar to existing pill organizers. .

Target Customer and Product:

The target customer for this product is both the retailer (pharmacy) and the consumer. First and foremost, the goal is to get pill sorting machines into pharmacies so that they may provide a new service to consumers. The service to be granted to pharmacies is the ability to supply consumers with their medication, all in one trip, in a sorted and pre-packaged manner. As mentioned in the need statement, one of the major problems with improving pharmaceutical drug adherence is simply getting people to pick up their medication from the pharmacy. By syncing up all of a patient's medication into one trip it ensures that only one trip is needed for their medication. Unfortunately, retrieving the medication from the pharmacy is only part of the problem. If someone fails to take their medication that could be harmful to their health but so can taking the medication incorrectly. This is why pharmaceutical medication must also be given to the consumer in a pre-sorted manner. By addressing both of these issues at the pharmacy, before the consumer receives their medication, pharmaceutical adherence will improve. Additionally, if the consumers opts for a medication dispensing device the provided packaging would be compatible with said device, further improving adherence by ensuring medication is taken properly. In the end by providing pharmacies with this opportunity it will further improve pharmaceutical sales while also improving the quality of life of the consumers.

Function Structure:

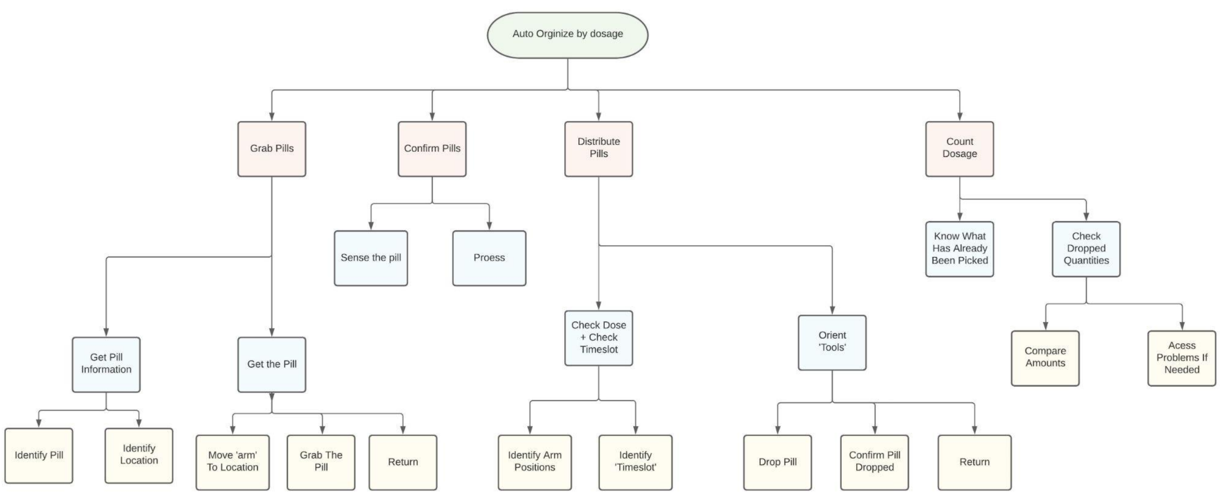


Figure 2: Function Structure

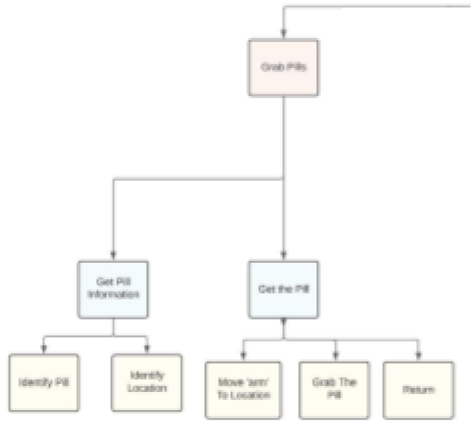


Figure 3: Function Structure - Grab Pills

Grab Pills

The first part of the process involves determining which pills are required for the current script. After gathering the itemized list from a user input, the device must first identify where the certain pills are being held in the machine. Following the confirmation, the machine will select one capsule. Once one pill has been selected it can be moved to a neutral position while avoiding other pills in the rare event the machine experiences failure.

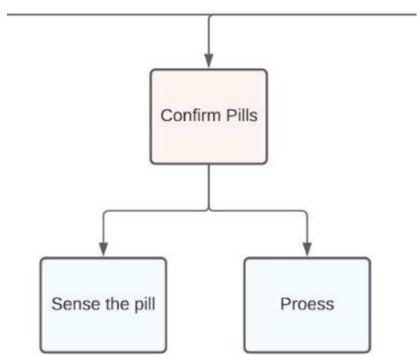


Figure 4: Function Structure - Confirm Pills

Confirm Pills

To ensure that the machine follows the input script perfectly, it is important to maximize the number of available redundancies. One error could catastrophically cause users to lose faith in reliability. After the selected pill has been retrieved it will go through a verification process. This could potentially range from simply confirming that a pill has been successfully grabbed to a trained computer vision program that would be capable of identifying subtle differences between prescriptions. Further testing will help determine the complexity of this step.

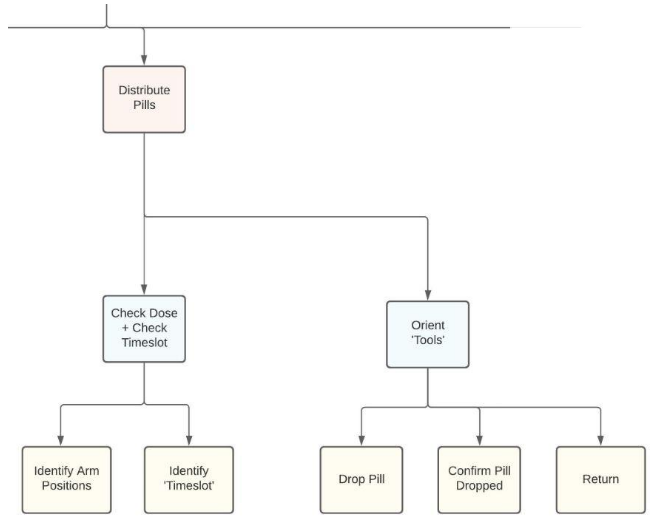


Figure 5: Function Structure - Distribute Pills

Distribute Pills

Before the machine can properly place the current pill into the correct distribution slot it needs to determine the drop off location by referencing the prescription. Once the coordinates are confirmed, the arm and/or distribution slot can be positioned for the determined drop off location. Once the pill is safely placed in the correct location, the machine can return to a neutral position, ready to retrieve the next required dosage.

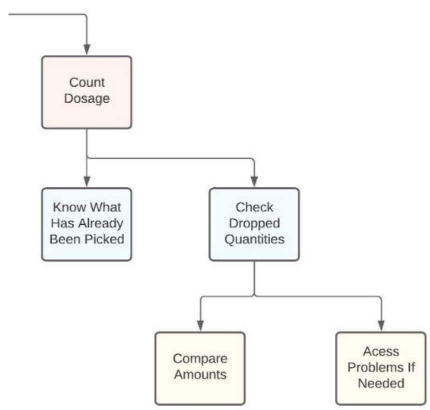


Figure 6: Function Structure - Count Dosage

Count Dosage

Finally, once the process has been completed, another redundant system will be utilized to check that the pill has been properly dispensed into the correct location. This will be done by referencing current data and seeing if it matches the system's assumed conditions. Similar to the Confirmation step, the complexity of this phase will be realized once future prototyping begins. If the real time data is in agreement with the system's the process can be continued. In the rare event uncertainty does occur, the program will be halted until an operator is able to remedy any issues.

In addition to the figure below of the function structure there is a larger version in a portrait view in the appendix.

Constraints:

As one might imagine, successfully delivering medication in a sorted and packaged manner comes with constraints that are physical and legal in nature. To begin, there are many physical characteristics that will bind our design due to both the nature of medication and the space of operation. Starting with the medication, *pill size, weight, shape, and texture are all important factors* that must be considered as they influence both the machine's mechanisms for sorting and total capacity. For example, a medication stored in a large gel capsule will have a much different shape and texture than a small powdery tablet making it imperative that the machine can easily handle not only these two types, but any type. The pictures below show some of many different capsule and tablet sizes used. These pictured pills are some of the most common shapes and sizes but there are many others such as those in the shape of an American football or just about any polyhedral shape.

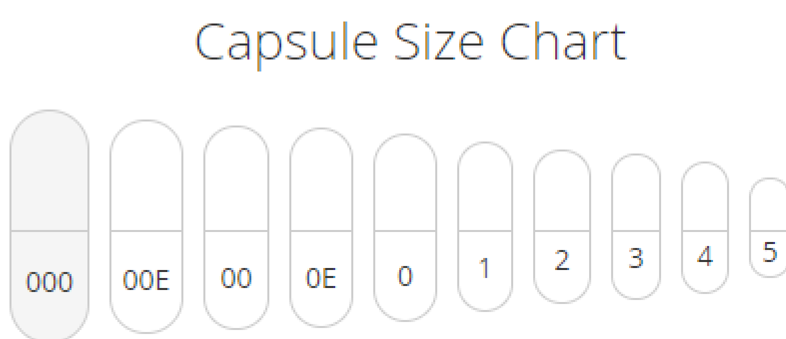


Figure 7: Capsules [2]

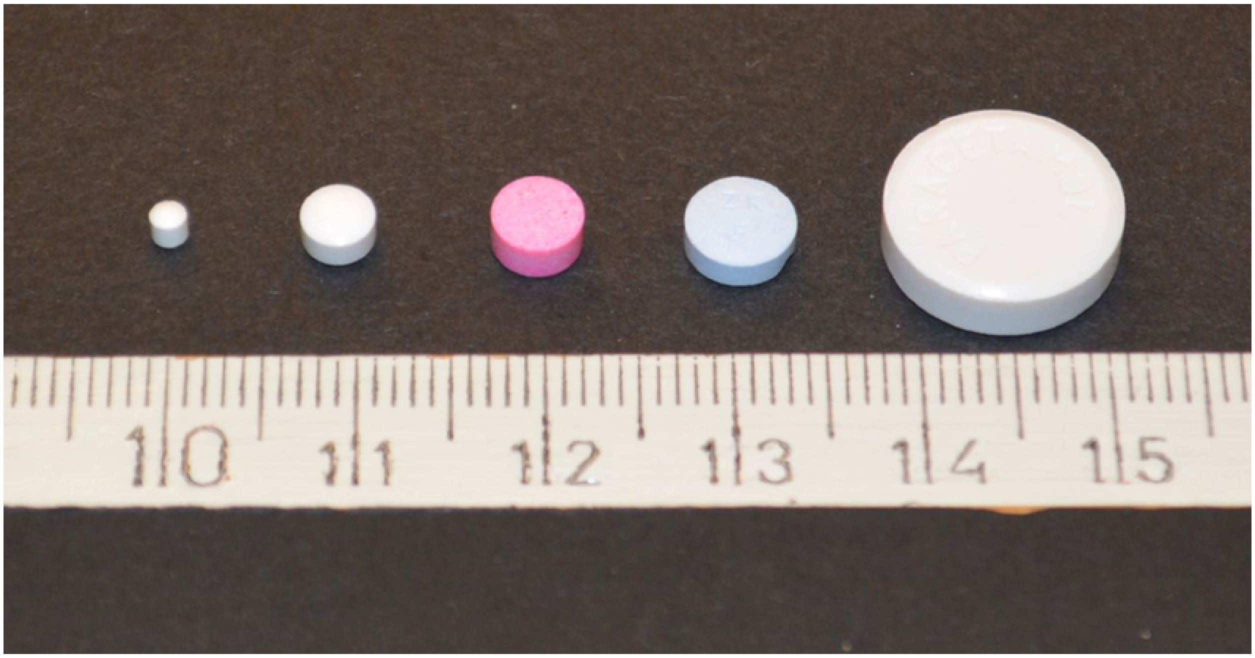


Figure 8: Tablets [3]

Another important factor to consider is the *machine's longevity as it could be lethal if it were to malfunction* during sorting. If the machine malfunctions during sorting it could distribute the pills incorrectly leading to potential harm to the consumer. It is also possible that the machine could malfunction during sorting and cause injury to the monitoring technician. This leads to the most important constraints which are the machine's need for a *high degree of precision and accuracy*. Even if the machine operates without malfunctioning, the machine is useless if it sorts incorrectly. This brings up potentially the most limiting constraints, which are the laws and regulations put in place by the FDA. An additional consideration to be made here is that sorting must occur in a timely fashion such that its speed matches or betters that of multiple people.

Order of Magnitude Estimations:

In calculating the value for a pill sorting machine in a pharmacy it is important to look at how the staff at a pharmacy would compare to a single pill sorting machine. To perform these estimations some assumptions need to be made about the pharmacy techs who gather the medications. The most important assumptions being the tech's hourly pay, script fill rate (the amount of scripts a tech can fill in an hour), the number of tech on staff at once, and the daily operating hours. One additional consideration that was made is that across the pharmacy technicians most likely are not actually working the entire time while on duty. With these pieces of assumed information, an hourly fill rate for the pharmacy can be found and the cost associated with operation. Located in the table below are the results of these calculations with the assumption being highlighted in blue and the calculations in orange. Next to these calculations are similar calculations for the proposed pill sorting machine. For these calculations the following is assumed, the initial cost, power consumption, and electricity cost. For the assumed power consumption similar mechanical machines were examined for reference such as automated vinyl cutters which use ~30W. The electricity cost assumption was obtained from the US national average for residential use so it is still higher than would be expected for commercial use.

Table 1: Value Estimations

Pharmacy		**Machine**	
Pharmacy Tech Hourly Rate ($/hr)	15	Cost ($)	500
Tech Fill Rate (scripts/hr)	20	Scripts to Fill at Machine Cost (scripts)	667
Labor ($/script)	0.75	Scripts per day to equal techs (scripts)	720
		Scripts per hour to equal techs (scripts)	60
Techs on Staff	3		
Combined Hourly Tech Rate ($/hr)	45	Power Consumption (W)	50
Total Fill Rate (scripts/hr)	60	Electricity Cost ($/kwh)	0.070
		Operating Cost ($/workday)	0.042
Daily Operating Hours (hrs)	12	Operating Cost ($/week)	0.294
Total Labor Costs ($)	540		
Actual Time Spent Working (hrs)	11		

With the value calculation out of the way the just as important force estimates remain. There are two methods in consideration for picking up the medication. The first being to use a simple pinching mechanism similar to a claw. The second method is to use a suction device, creating a difference in pressure between a small tube and atmospheric pressure letting the pill be held. To begin, let's first look at method one. Located in the figure below is a force diagram of this configuration.

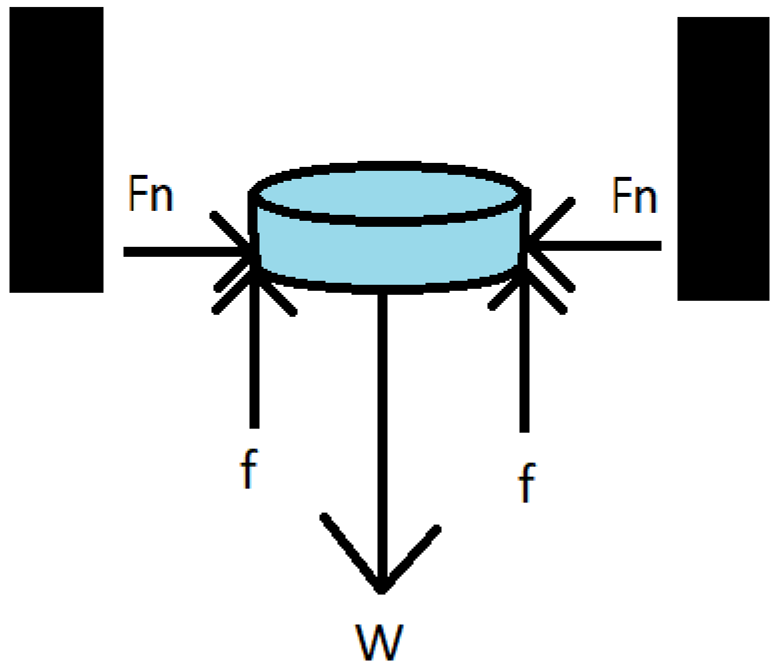


Figure 9: Force Diagram - Grabbing

Using this method the pill is held in place through a frictional force which is proportional to the normal force produced by the pinching mechanism. For these calculations the coefficient of static friction is assumed to be similar to that of the interaction between polystyrene and steel. Additionally, it is assumed that the maximum pill mass will be 5000mg so the amount of weight to be lifted is easily calculated. From these assumptions the normal force required to hold the maximum size pill is obtained. Another consideration to be made using this method, and any method, is that care must be taken when grabbing that pill so that they do not break. To obtain an estimate of the minimum force required to break a pill, a value for a pill's ultimate strength is assumed. For this case the value is assumed to be half that of an alumina porcelain which is 55 MPa. Using the assumed ultimate strength and the smallest pill diameter of 1.5mm, the force required to break a pill can be calculated. All of these assumptions and calculations can be seen in the table below.

Table 2: Grabber Forces

Forces	
Max pill weight (mg)	5000
Coefficient of Static Friction	0.3
Smallest Pill Diameter(mm)	1.5
Compressive Strength (MPa)	27.578
Smallest Area (mm^2)	1.767
Max Gravitational Force (N)	0.049
Force to Grip (N)	0.082
Minimum Force to Break (N)	48.734

Now to consider the second method of using a suction tube to grab the pills. A force diagram of this configuration is located in the figure below.

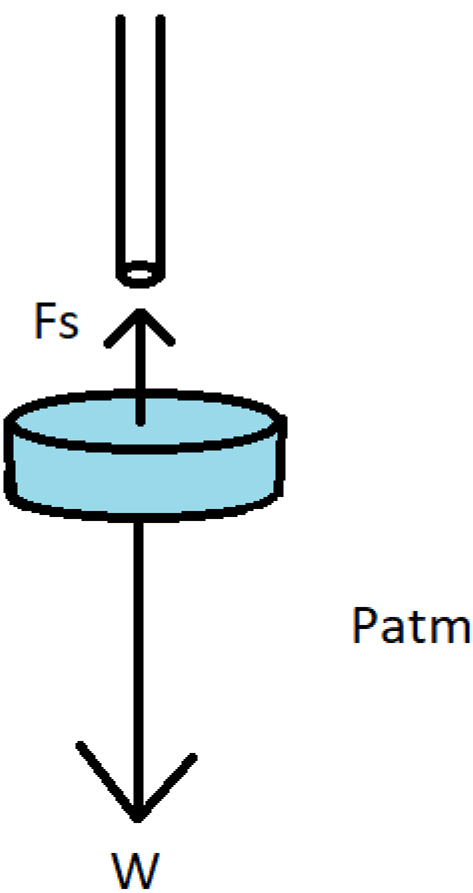


Figure 10: Force Diagram - Suction

Using this method there are only two new assumptions that need to be made to obtain the necessary calculations. The first is that operation will be occurring at standard atmospheric pressure and the size of the suction tube will be 1mm in diameter. Using these values the amount of force required to break suction between the pill and the tube can be calculated. The results of these calculations can be found in the table below. In performing these calculations it is essential that the force to break the suction be greater than the gravitational force on the pill. Using the largest assumed pill mass of 5000g it is found that there is a gap between these values of roughly 37% making this method feasible.

Table 3: Suction Forces

Forces	
Atmospheric Pressure(KPa)	101
Suction Diameter (mm)	1
Max Pill Mass (mg)	5000
Max Gravitational Force (N)	0.049
Suction Area (mm^2)	0.785
Force to break suction (N)	0.079

Requirements:

Performance	• Accommodate tablets 1.5mm to 15mm in diameter
	• Accommodate capsules 11mm to 26mm in length
	• Pill mass ranging from 1mg to 5000mg
	• Functions for varying pill textures
	• Must maintain accuracy and precision throughout operation
	    ○ Accuracy of 99.99%
	• Fast enough to fill 60 prescriptions per hour
	    ○ 1800 pills per hour
	• Grip force greater than 0.1 N

Size	• Fit within 600mm x 600mm x 600mm space • Weigh less than 20kg • Function on 110V AC power
Safety	• Adhere to FDA, DEA, and EPA regulations • Incorporate proper safety measures to prevent workplace accidents.
Value	• Prototype cost under $500

	● Have a weekly operating cost of less than $1 ● Intuitive operating system requiring little training ○ Pharmacist training time of less than 1hr

III. Conceptual Design

Introduction

As has been outlined in the need analysis section of this report, there are many difficulties and constraints posed by this project. Firstly, size is a major problem as the entire machine should be able to fit through a standard 36” x 80” door frame. Outside of getting the machine into a room the machine will need to be able to reliably move medications from one location to another without damaging the medication. To go along with this, there is no margin for error in the sorting of medications into correct bins as one misplaced medication could lead to a death. Additionally, with the moving of medication the machine will need to be able to accommodate a wide range of pill shapes and sizes. This leads to the requirement of multiple redundancies in the final design to ensure the safety of the consumer. Finally, as examined in detail in the function structure section within the need analysis, the four top level functions of this machine are to grab pills, confirm pills, distribute pills, and to count pills.

Concept #1

A notable machine, which has many parallels to this project’s goals, is a SMT pick & place used for printed circuit board (PCB) production. These types of machines are able to select components from a wide range of available parts. The mobile axis also allows for the inclusion of visual identification and configuration through computer vision and machine learning.

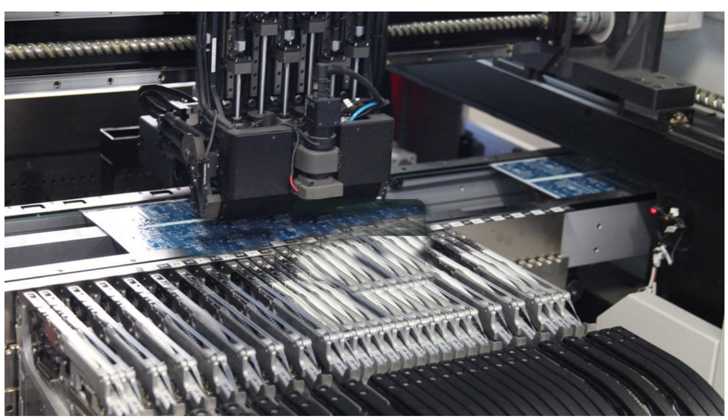


Figure 11: Pick and Place machine

However unlike a typical pick and place machine used to populate PCBs, a pill sorter will not need the same XY axis resolution. Instead, the pills will be dropped into small bins. This dramatic decrease in distribution precision allows us to remove an axis to simplify the design. The style allows for easy expansion to accommodate larger quantities. Since there is only one major axis, it would be very easy to increase the axis beam length and install longer belts. A larger work surface would also provide ample space to integrate the necessary sensors to provide redundancies.

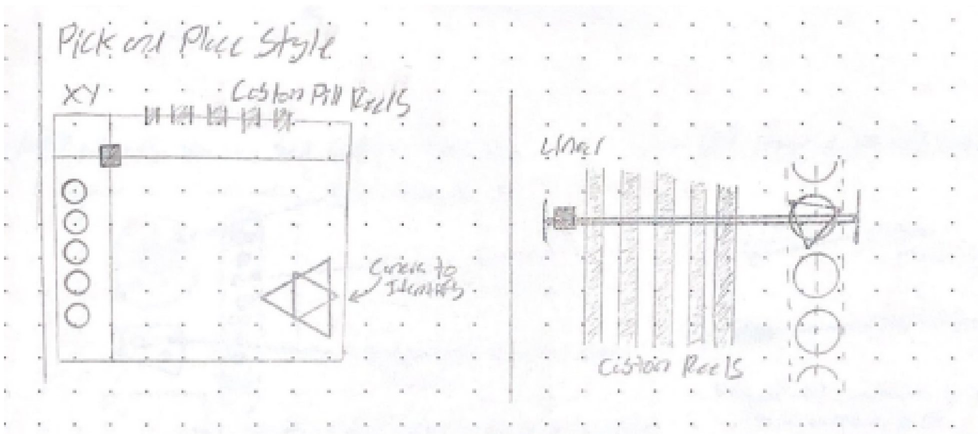


Figure 12: Initial Concept: One vs. Two axis design

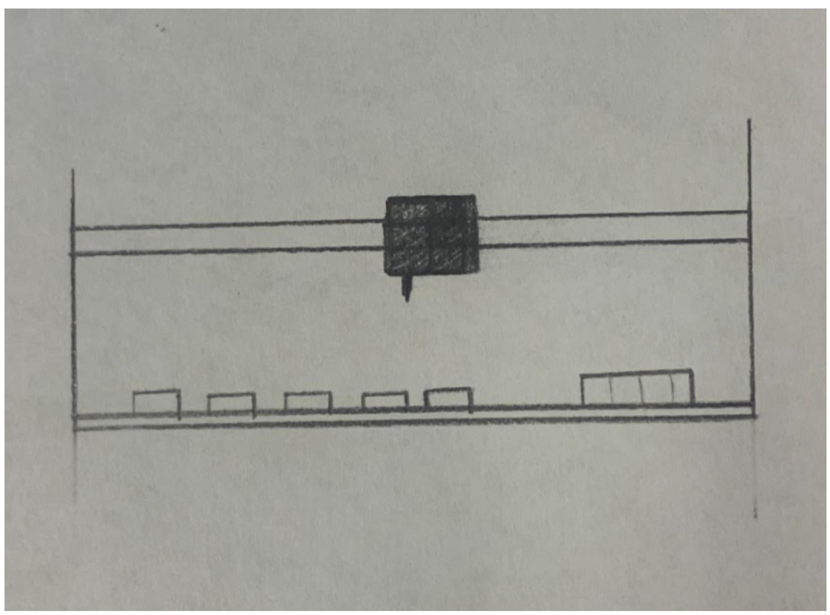


Figure 13: One axis design

In preparation for sorting, each pill type could be stored together in individual housings. The housings will act as a slide to ensure that there is always a pill available to the picker with it’s limited degrees of freedom.

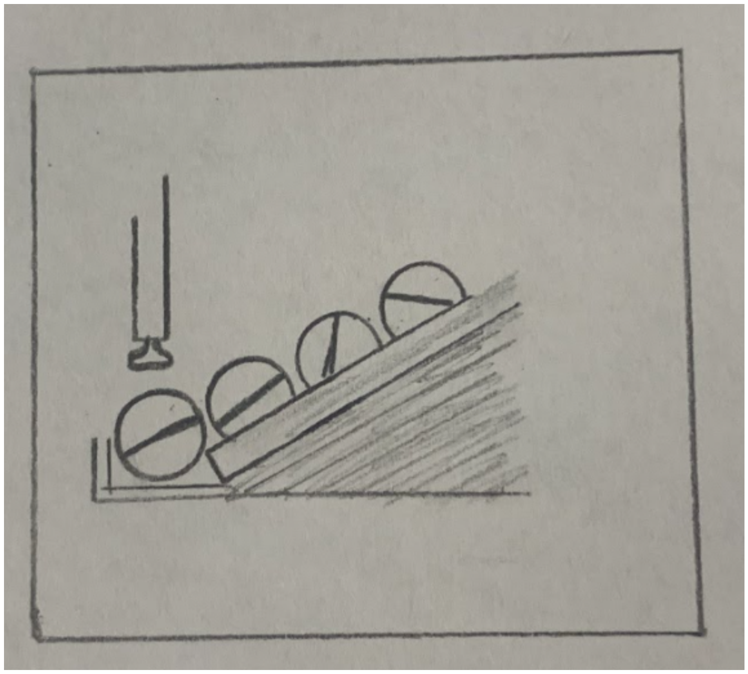


Figure 14: Pill “slide”

The picker assembly will be attached to a linearly moving, belt driven chassis. A counter weighted Z-axis belt design will allow the picker to grab and place pills while avoiding obstacles. The counterweight system has many advantages, most notably the space for a second device. This could either be another picker to increase sorting times or a camera to increase accuracy, depending on the system’s needs.

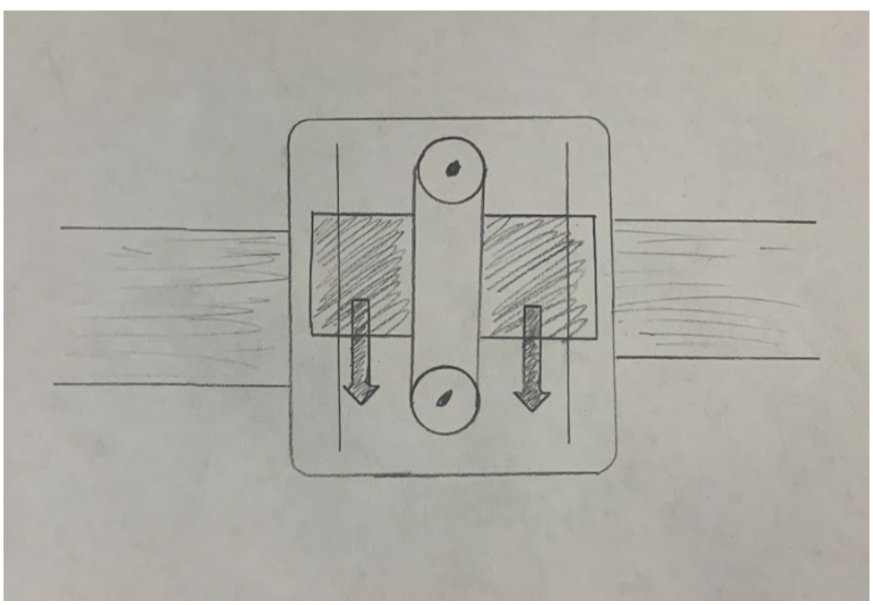


Figure 15: Counter weighted Z - axis assembly

Once the pill has been grabbed, It will be sorted into predetermined ‘bins’ that represent various time intervals throughout the week. These bins will be arranged in a cylindrical style on another stepper controlled platform. The platform will rotate, resembling a record player, providing the picker head the ability to successfully travel to each 'bin'.

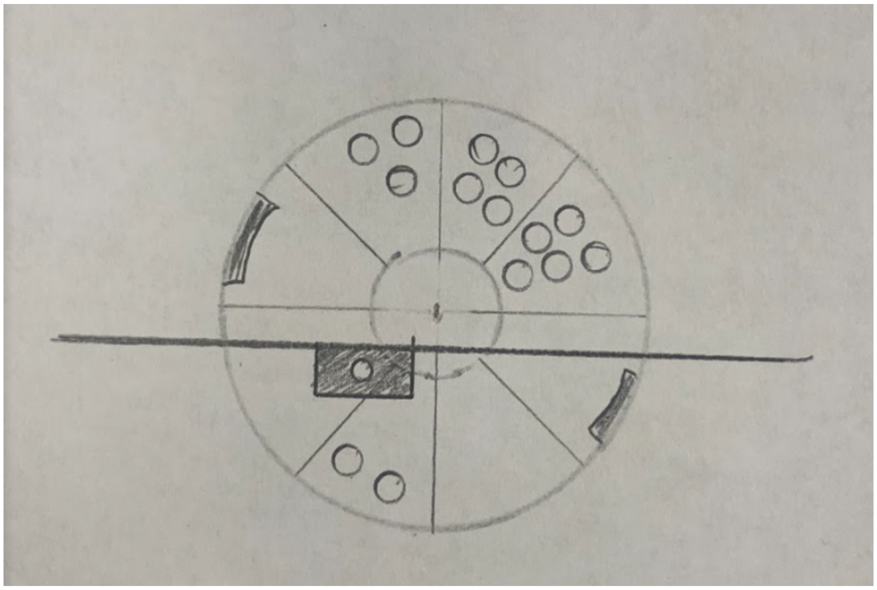


Figure 16: Sorted Pill Box

For further stages of this project, it would be very easy to design a basic assembly line to automatically exchange 'bin' housings.

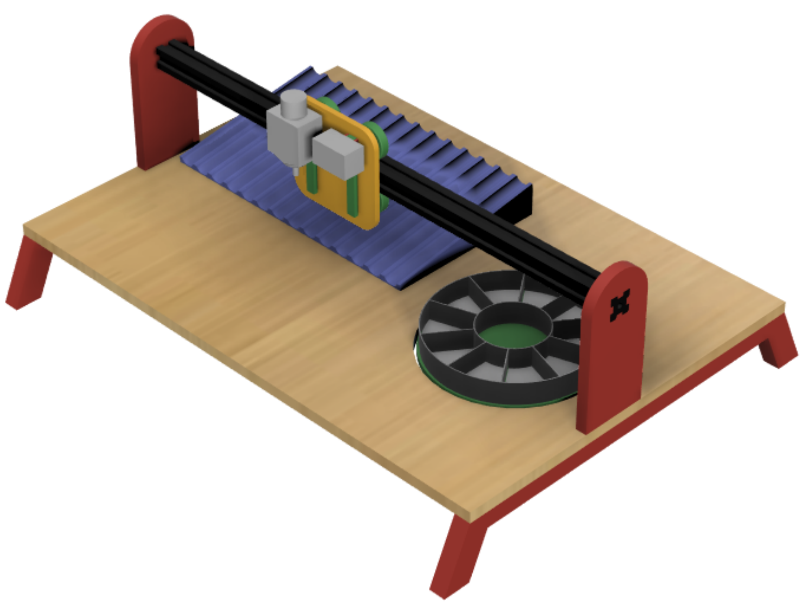


Figure 17: Complete Assembly CAD Concept

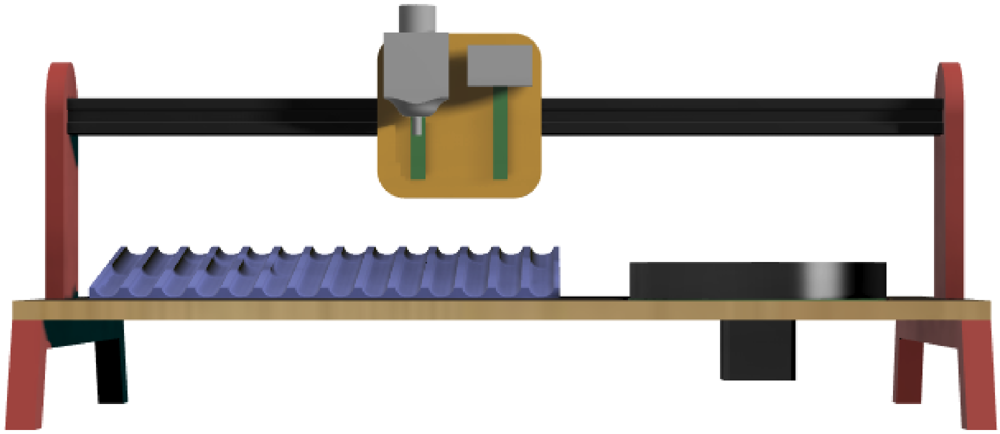


Figure 18: Complete Assembly CAD Concept Front

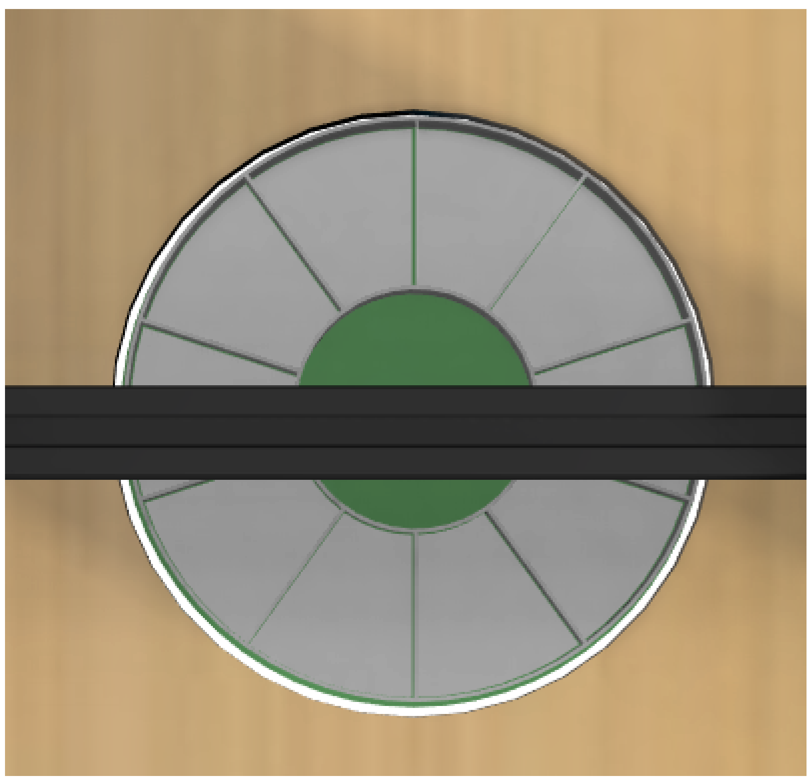


Figure 19: Pill Sorting Bin System

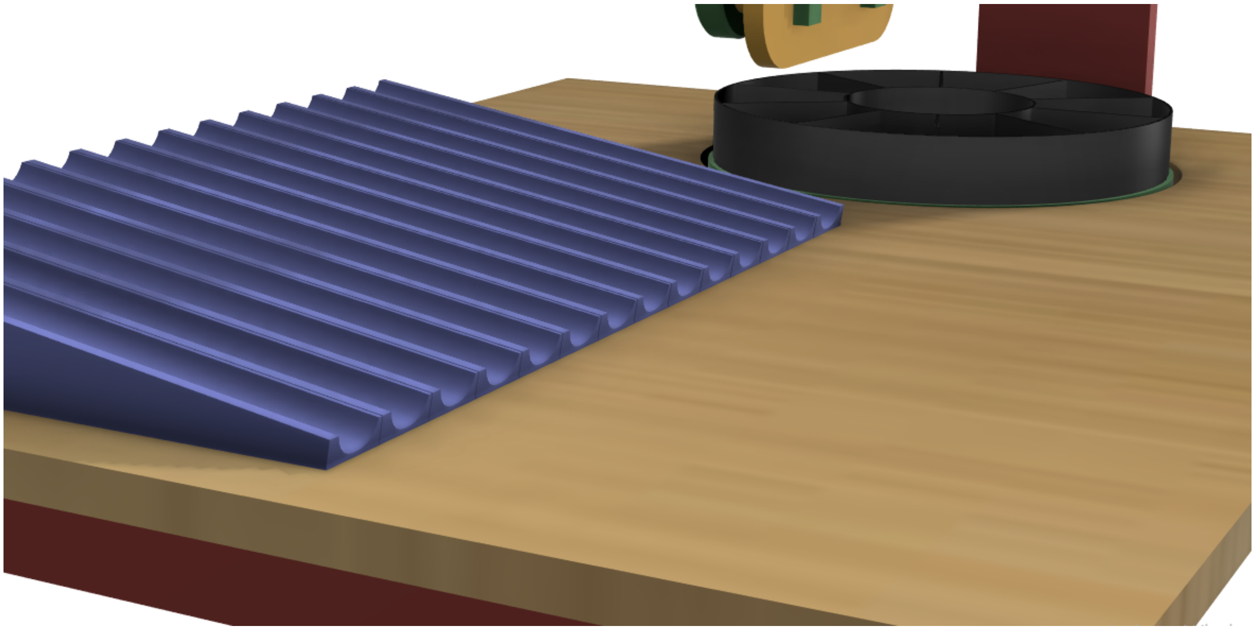


Figure 20: Pill Sorting ‘Slide’ Concept

It was decided each sorting ‘bin’ should be capable of holding 15 maximum size capsules (000) creating a volume per division of ~8400 mm^3. Using this, paired with the cylindrical design, the minimum diameter of the container is found to be 61.3 mm. To allow for some extra wiggle room, the space was increased to an 80mm diameter.

Table 4: Sorted Pill Box Sizing

Sorted Pill Container	
Pill per Division	15
Volume per Division (mm3)	8440.35
Total Volume	59082.45
Minimum Diameter (mm)	61.3
Total Weight (kg)	0.173

The most strenuous mechanical part of the assembly will be the picker head. Not only will it have to quickly travel along the x-axis, it also needs to move the picker itself up and down to grab and release the pills. A standard Nema 17 stepper motor weighs roughly 400 grams. This will most likely be the source of locomotion for all axes. One will be attached to the head itself to control the counter weight. The counter weight Z-axis system consists of a belt attached to two linear guides. 40mm linear guides weigh approximately 20 grams each and the belt weight is neglectable. The most important part of the head is the picker itself. This is estimated to be 200 grams. Since we are using a counter weight system there will be another additional 200 grams on the other side of the system. Finally the rail mount, consisting of both the v-slot wheels and backplate are expected to weigh under 150 grams. This accumulates to a total head mass of 990 grams.

Table 5: Picker Head Weight

Central Unit Weight	
Stepper Motor (g)	200
Linear Guides(g)	40
Picker (g)	200
Counter Weight(g)	200
Rail Mount (g)	150
Total (g)	990

Concept #2

For this concept an SMT pick and place head will be used to retrieve medication that is stored in bins. In this design the pill bins will be stored in a circular arc around the SMT head in two levels allowing the head to access the pills by rotating to the correct position, moving to the correct level, and extending/retracting the SMT head for retrieval. To handle the sorting of the pills a circular pill box will be placed into a corresponding servo with a registration mark to ensure the same alignment every time. While retrieving a pill from the appropriate bin the servo/pill container will rotate the container into the correct day for the pill to be dispensed into. Located in the figure below is the initial concept.

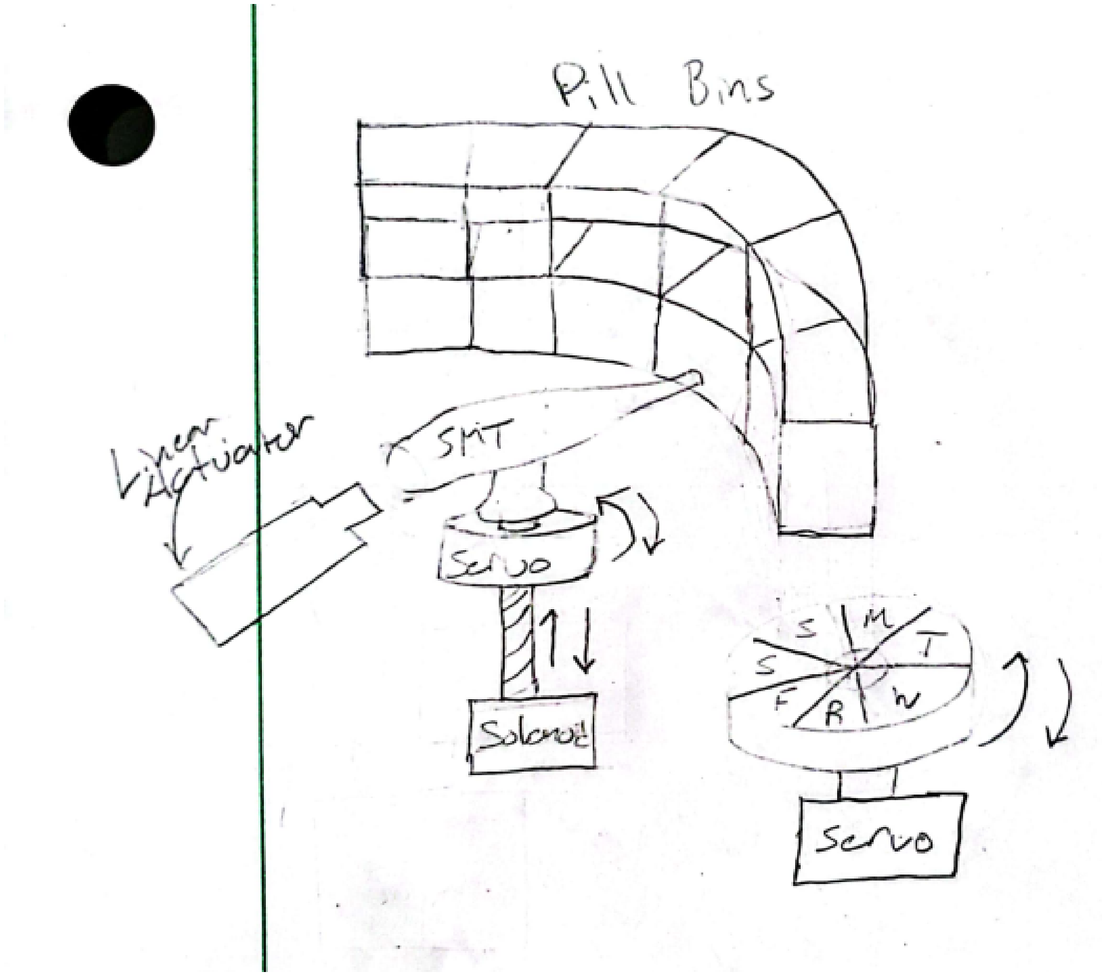


Figure 21: Initial Concept

The SMT head itself will sit atop a linear bearing that is guided by a set of guide rails to allow for extension and retraction of the head into the pill bins. In order to perform the task of extension and retraction of the SMT head a linear actuator will be used to allow for fine tuning in order to prevent damage to medication. To allow for the head to move between levels, the head will be attached to a solenoid with guide rails. Finally, to rotate the SMT head all of the movement mechanisms above will sit atop a servo.

As for the pill boxes themselves they will be manufactured so that the pills will gravitate toward the center of the box the same as a normal hopper. In order to fill the pill bins a slot is positioned at the back of the box for a technician to deposit the pills into. Now due to the fact that the smallest pill size that will be encountered is 1.5 mm in diameter and the previously calculated SMT head diameter of 1mm will be sufficient it will not be possible for the mechanism to grab more than one medication at a time. Additionally, the SMT head will be modified so that it has some give and will act as a gasket between the head and pill to improve pill suction. Located below are several figures looking at the individual components in more detail.

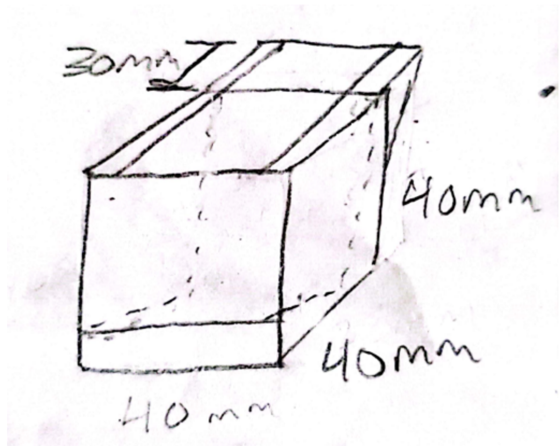


Figure 22: Individual Pill Box

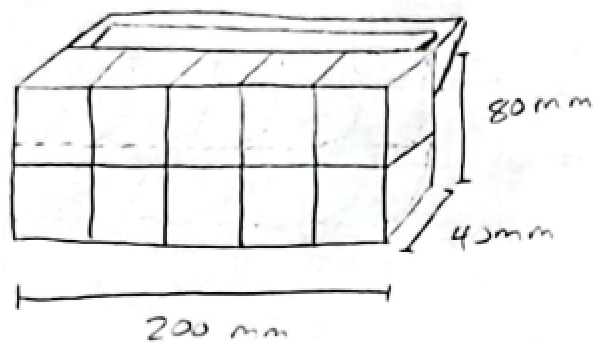


Figure 23: Pill Boxes

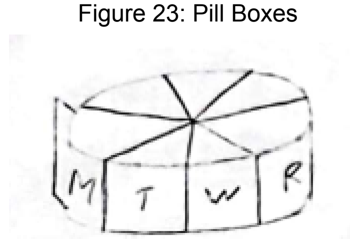


Figure 24: Sorted Pill Box

Figure 25: Sorted Pill Box & Servo

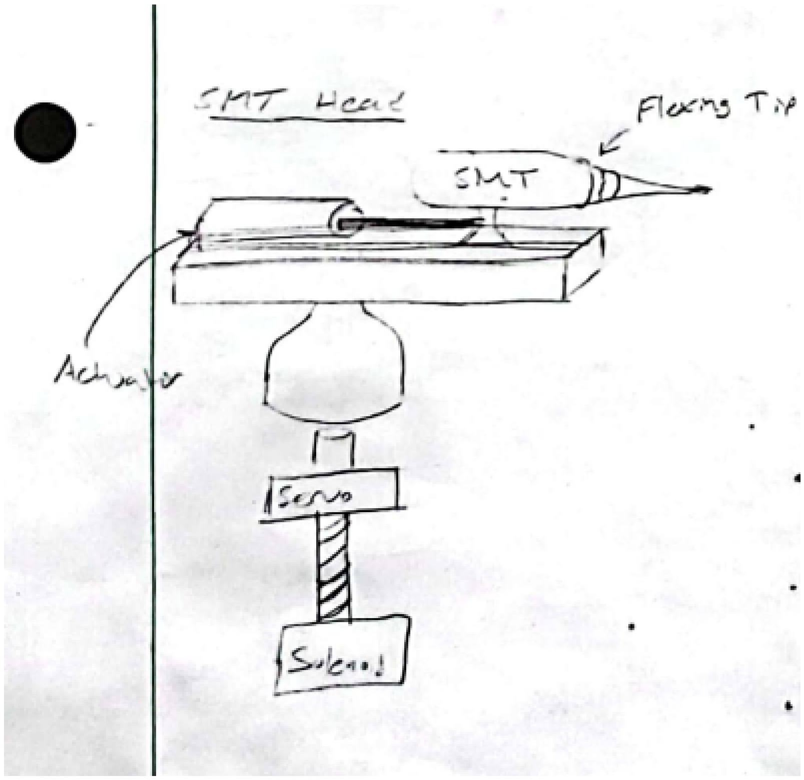


Figure 26: SMT Head Unit

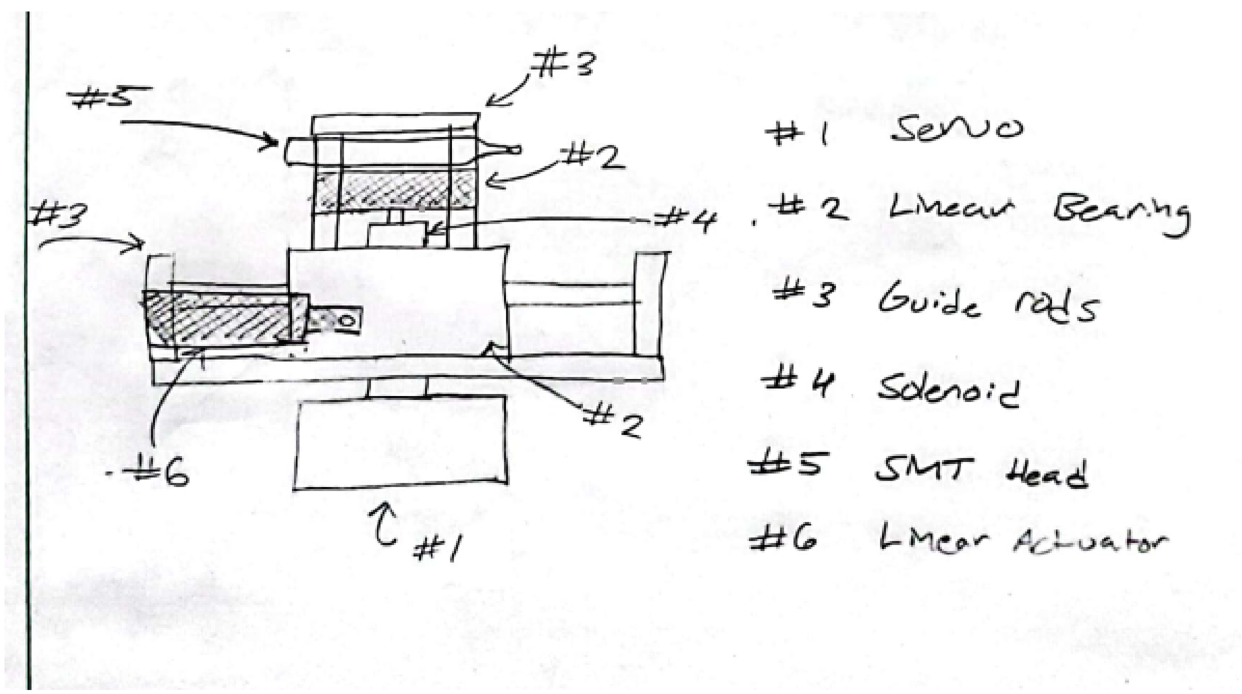


Figure 27: SMT Head Unit Diagram

In combining all of the above drawings and their respective dimensions the concept is estimated to easily fit with a 20 cm x 20 cm x 20cm space. This estimate is quite generous to account for all necessary power and control electronics as well as associated wiring.

From these above images and using rough dimensions for the SMT head, servos, and linear actuators the following model and its components were produced.

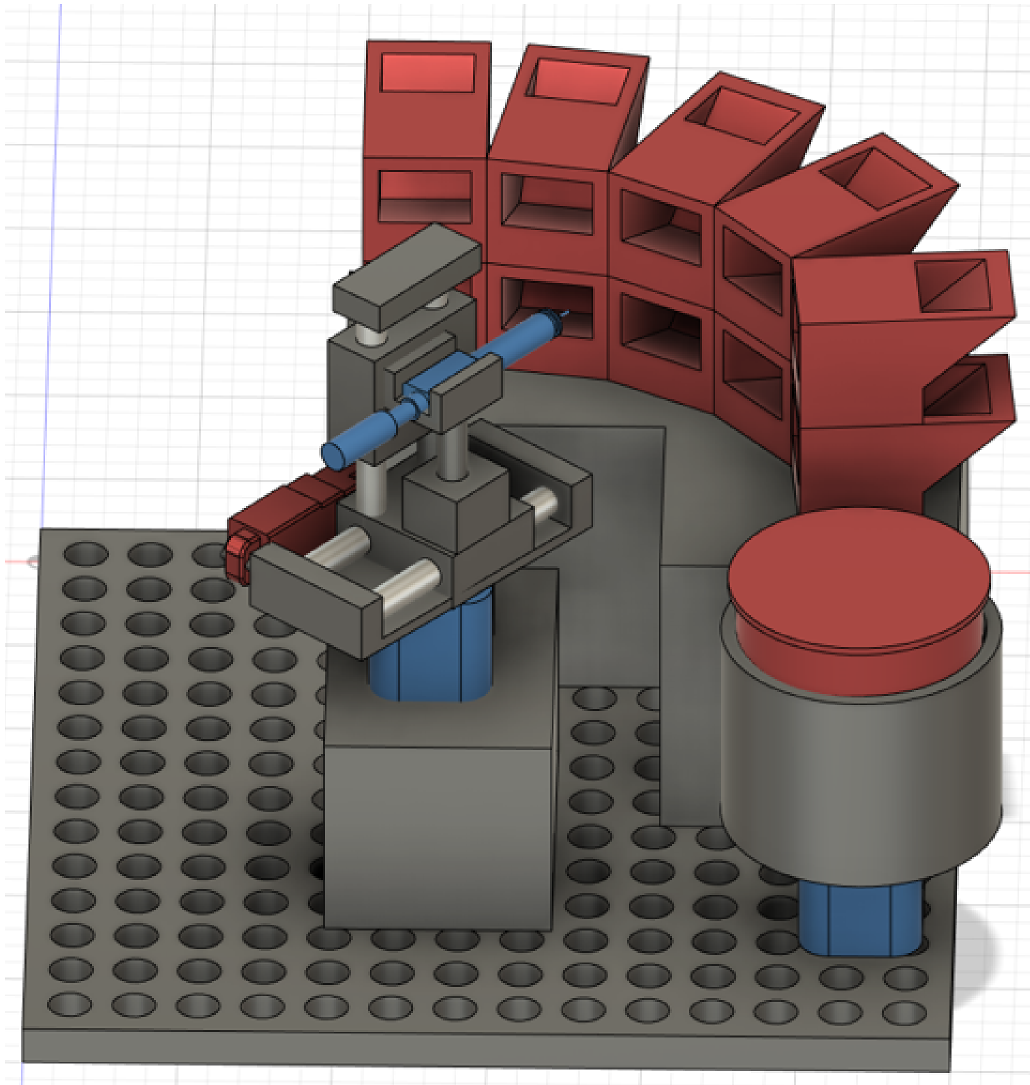


Figure 28: Complete Assembly

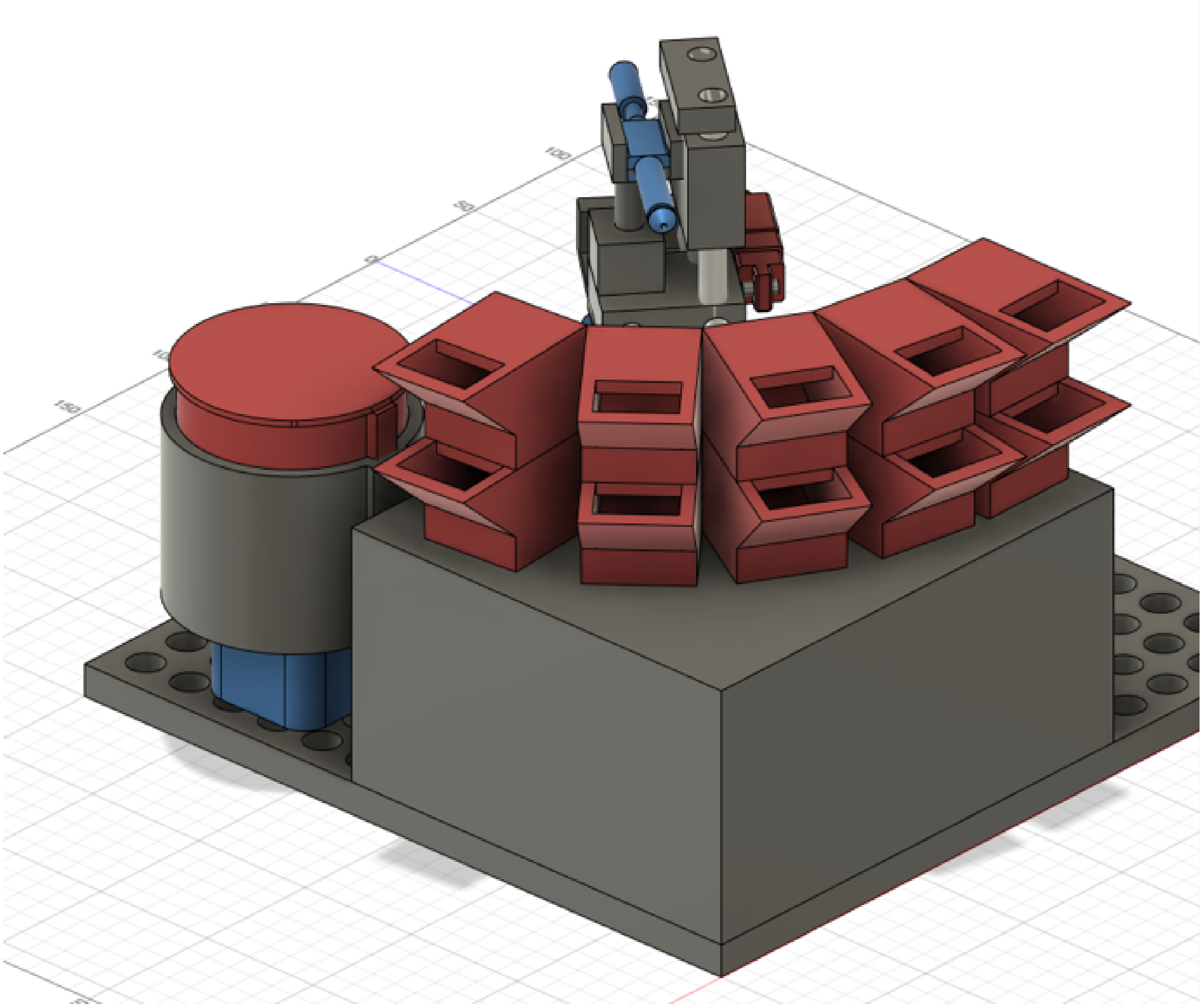


Figure 29: Complete Assembly

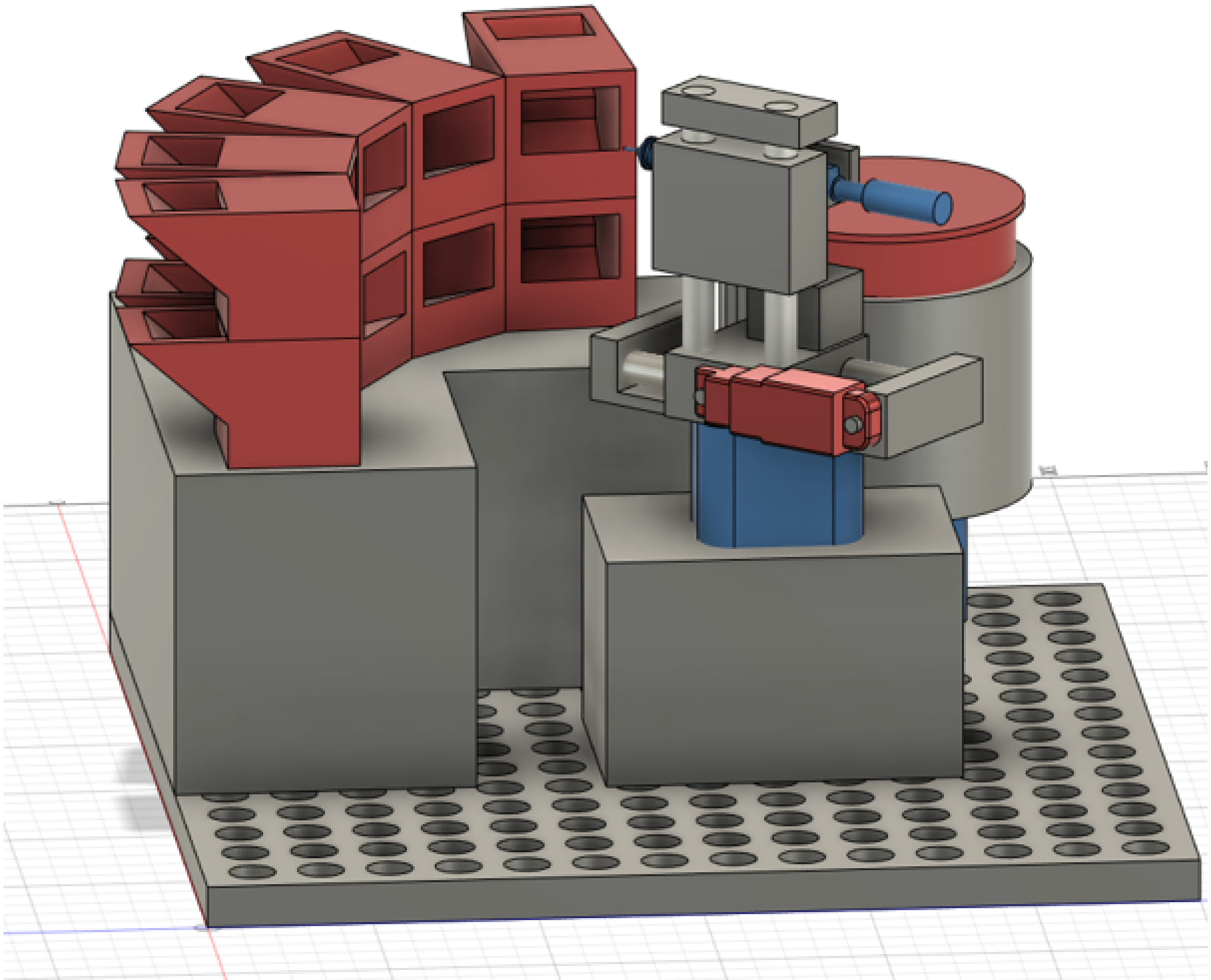


Figure 30: Complete Assembly

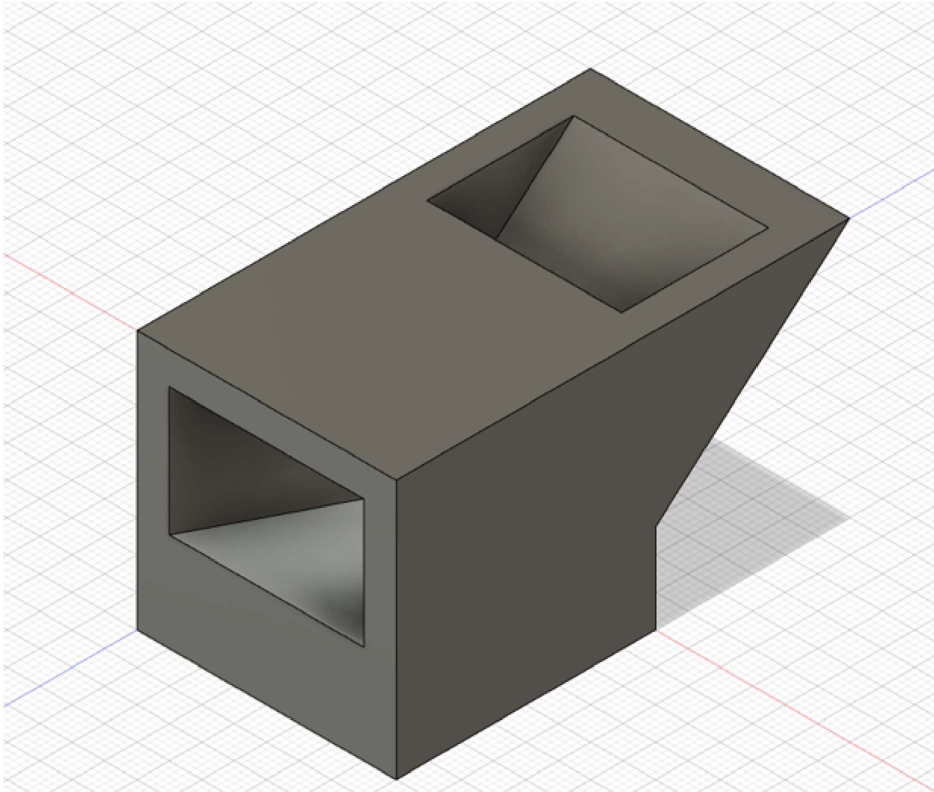


Figure 31: Pill Box

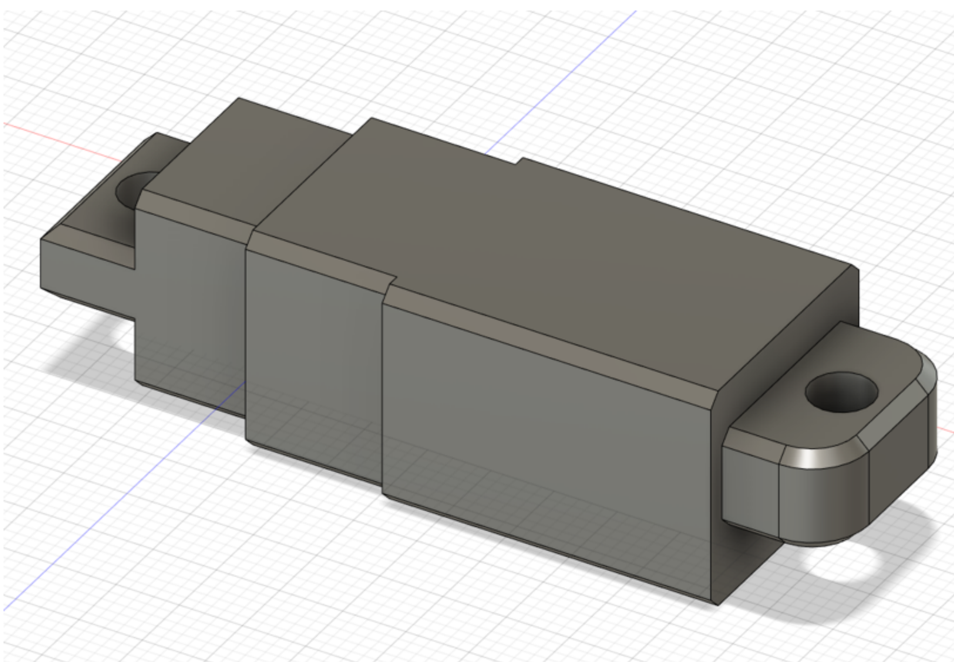


Figure 32: Linear Actuator

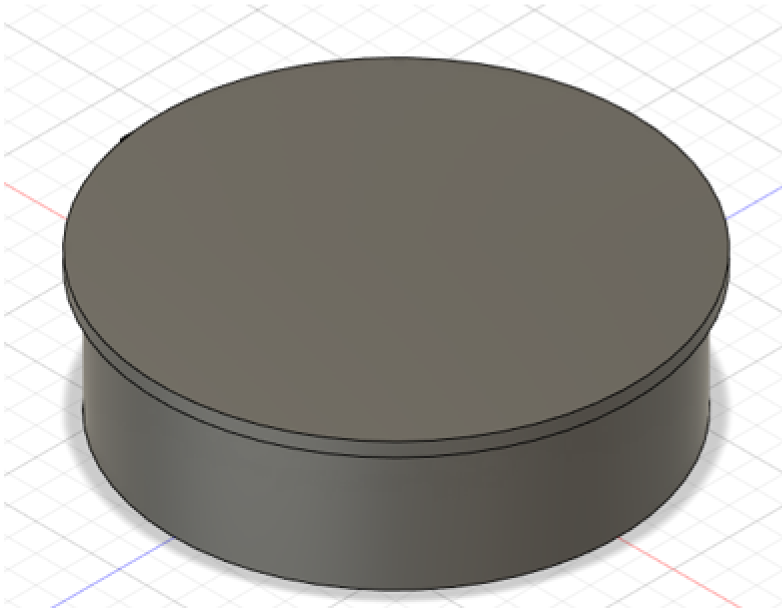


Figure 33: Sorted Pill Box

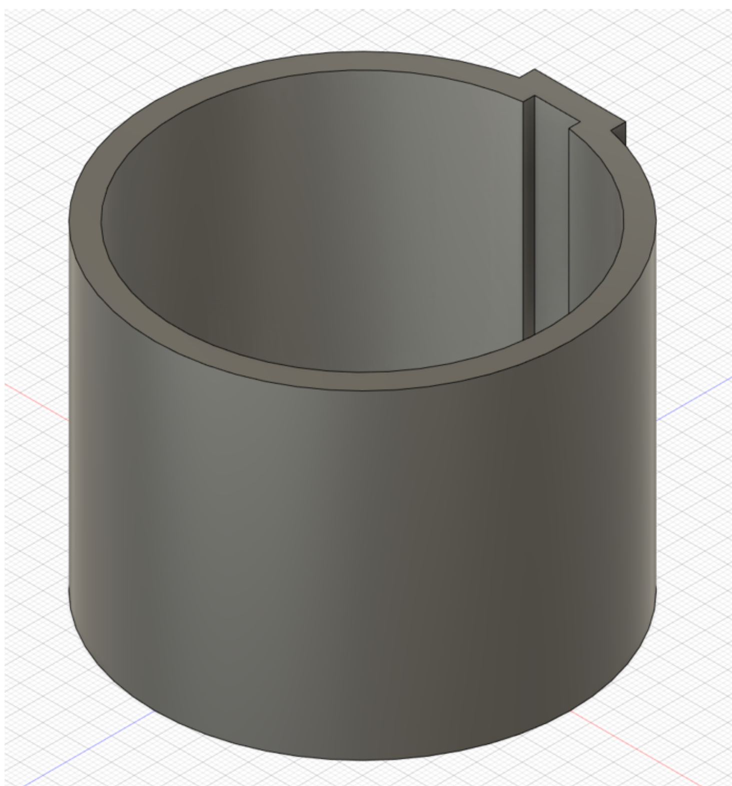


Figure 34: Sorted Pill Box Interface

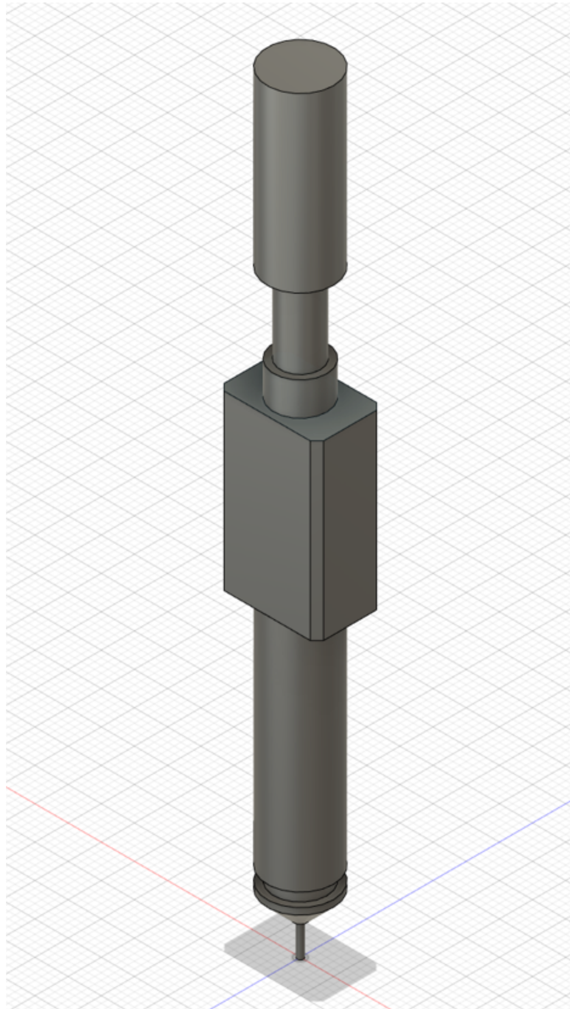


Figure 35: SMT Head

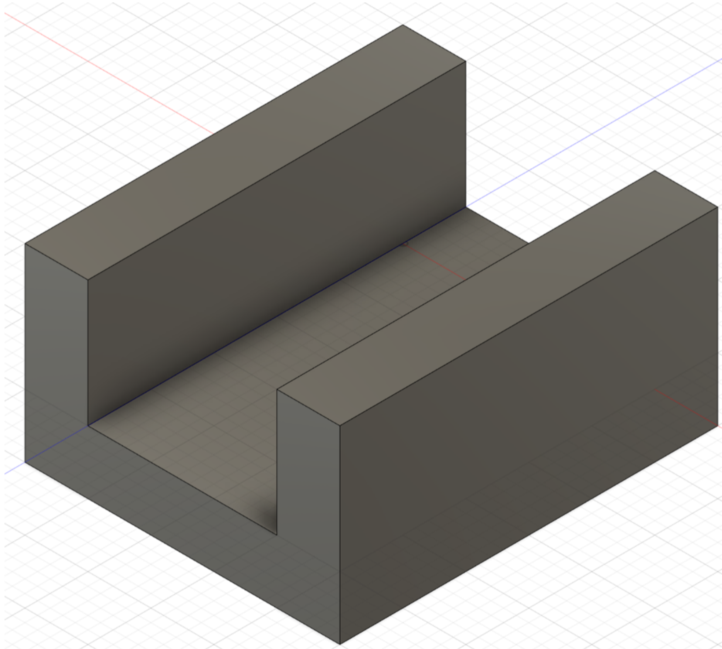


Figure 36: SMT Hanger

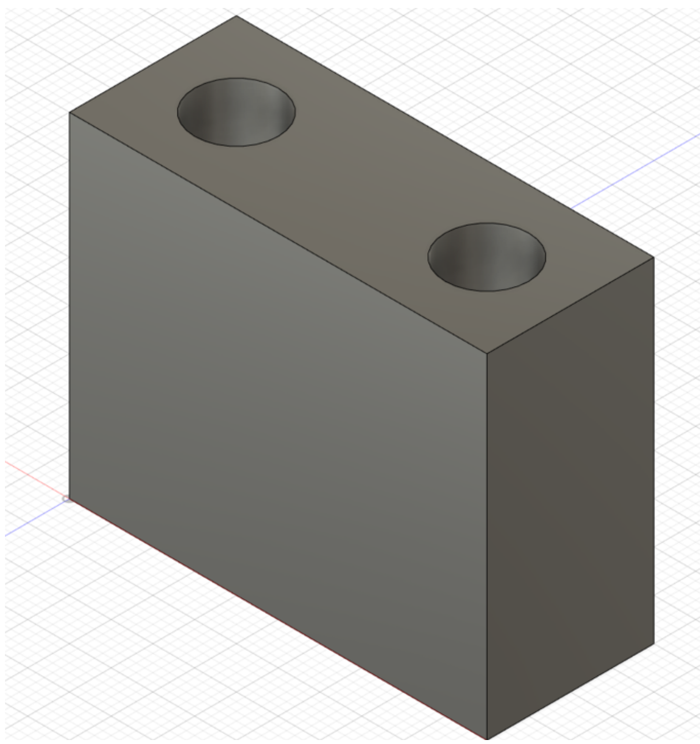


Figure 37: Linear Bearing

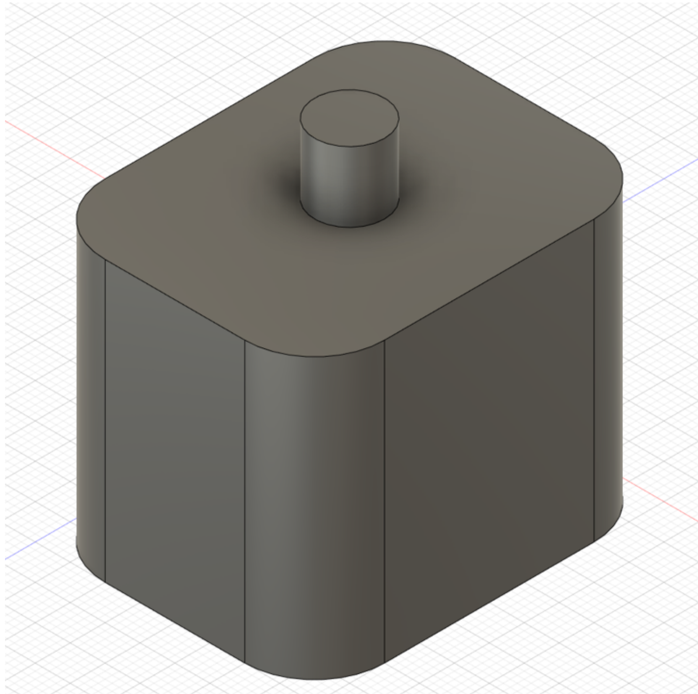


Figure 38: Servo

The majority of the calculations in this concept are concerned with the sizing of the pill boxes/sorting container as well as the requirements of the linear actuator, solenoid, and servos. First off is the pill boxes, which were sized by looking to match a volume similar to a large prescription pill bottle holding the largest 000 capsules. The pill bottle used for comparison was 40 mm in diameter and by using the standard size for 000 capsules resulted in a cube that is 40mm x 40mm x 40mm which equates to a pill storage of roughly 114 pills per box. As for the sorted pills container it was desired for each day of the week to be capable of holding 15 max size capsules (000) giving a volume per division of ~8550 mm^3. Using this and the desired shape of circle the minimum diameter of the container is found to be ~ 61 mm. Since not all of that internal space will be occupied by just pills the size of the container was sized up to a modest 80mm in diameter. A summary of these calculations can be seen in the table below.

Table 6: Pill Box Sizing

Pill Box Sizing			
Large Pill Bottle (mm)	40	Max Pill Size	
Volume Pill Bottle (mm3)	75000	Mass (mg)	1644
		Length (mm)	26.1
Volume 40mm cube (mm3)	64000	Lcap (mm)	12.95
Max Pills	114	Volume (mm3)	563

Table 7: Sorted Pill Box Sizing

Sorted Pill Container	
Pill per Division	15
Volume per Division (mm3)	8550
Total Volume	591005
Minimum Diameter (mm)	61
Total Weight (kg)	0.175

Now for the sizing of the mechanical components the most important mechanisms are the central servo controlling SMT head rotation, the solenoid controlling level selection, and the linear actuator controlling extension and retraction. Starting off with the central servo it must be determined the amount of weight that it will be required to function under so the components weights atop it must be summed. First off the SMT head itself will weigh roughly 200 grams. The linear bearing will weigh ~100 grams a piece giving a total mass of 400 grams. The linear actuator will be roughly 100 grams. The guide rods themselves along with any additional mounts will be material dependent so they will be assumed to weigh 200 grams. This gives a combined mass of 900 grams that the servo will need to be able to rotate. This equates to a force of roughly 9 N that the servo will need to be able to rotate. The calculations are summarized in the table below.

Table 8: Central Unit Weight

Central Unit Weight	
SMT Head (g)	200
Linear Bearings (g)	400
Linear Actuator (g)	100
Guide Rods + Mounts (g)	200
Total (g)	900
Force (N)	9

In addition to the central unit servo there is the servo for rotating the sorted pill container. For this unit there is the weight of the container interface, the container, and the pills. The assumed mass for the interface is 100 g and the container is 50 g. The weight of the pills is variably, but at max load which is when every pill division is filled with max size capsules the total weight of all the pills is 1265 g. This gives a total mass on top of the servo of 1415 g. These numbers are summarized in the table below.

Table 9: Sorted Pill Unit Mass

Sorted Pills Weight	
Container (g)	100
Interface (g)	50
Pills (g)	1265
Total (g)	1415
Force (N)	13.8812

One final piece to look at is the required extension retraction range of the linear actuator. Assuming that the pill boxes are arranged in a perfect quarter circle around the SMT head the minimum inner radius that the SMT head may sit at is 130 mm and the outermost point would be 170 mm inside a pill box. The summary of obtaining these numbers can be found in the table below

Table 10: SMT Head Range

SMT Head Range	
Arc Length (mm)	200
Angle (deg)	90
Inner Radius (mm)	130
Outer Radius (mm)	170

Now comparing this to the components that were used in sizing the model it is seen that the components used in the model may be overkill but the components will provide the speed required to be better than multiple humans. For example, the linear actuator used has a quoted extension/retraction speed of 150 mm/s which will meet the project's demands but the design only will require the SMT head to extend/retract ~40 mm.

Concept #3

For this design a 3-axis servo motor driven SMT head will be maneuvered around a bed of pill bins. Before going into further detail about this design please see the sketches below to get a visual representation of the design.

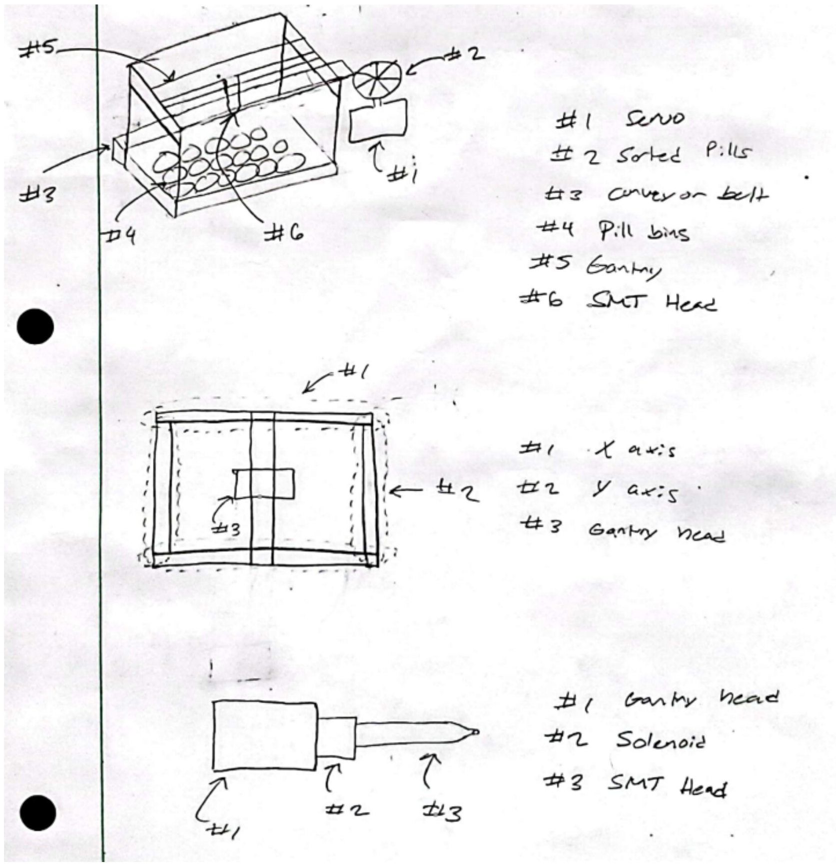


Figure 39: Initial Concept

In the pill bin bed there will be individual sections for each type of pill that taper to the bottom as a type acting as a hopper to ensure pills are centered. In order to position the SMT head into place over the correct locating a 2-axis gantry will be used where the x and y axes are driven by servo motors attached to belts. In order to keep the gantry in line it will be guided by sets of guide rods fitted with linear bearings that will be attached to the belts that the servo motors are driving. Once the gantry head is in position a solenoid piston valve will activate causing the SMT head to descend into the pill bin where then the SMT head will grab a pill. Once a pill has been retrieved from the bin the gantry will move to the closest position over a conveyor belt to deposit the pill. Once the pill is on the conveyor belt the pill will be transported down to a servo controlled sorted pill bin and deposited into its corresponding bin. The servo and sorted pill bin accomplishes this by rotating the pill bin to the correct position prior to the pill dropping into the bin.

In choosing the sizing of the pill bins a large cylindrical pill bottle with the dimensions of 40mm in diameter and 60mm in height was used as reference for the desired sizing of the bins. When performing the calculations for his pill bin size the amount of pills possible to be stored per bin is found to be well over 113 pills which is much more than necessary. The other import vessel here is the container for the sorted pills which was desired to be capable of holding 15 max size (000) capsules per day of the week. When performing the calculations to determine the size required for this vessel the minimum required diameter is found to be 61.3 mm at a height of 20mm. As can be seen these are the same sizing that was used for the previous concept so please see the tabulated data in the previous section for these numbers. For the final size of this concept the machine is expected to fit within a 20cm x 30cm x 20 cm cube quite easily due to the expanded spatial requirement with a conveyor belt.

For a better understanding and visual representation of this design see the picture of the 3D generated model below. This model was generated using SolidWorks Fusion 360.

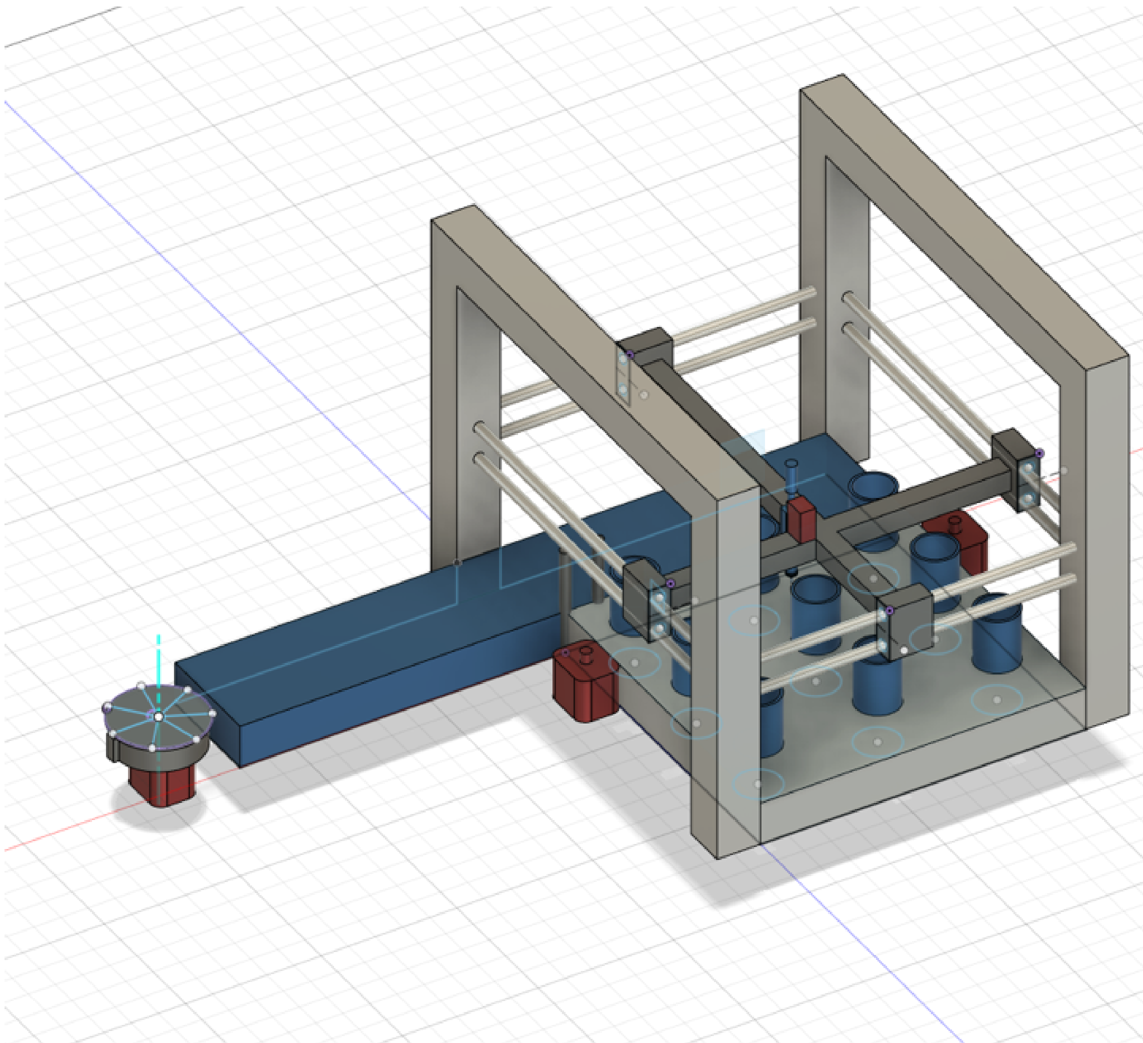


Figure 40: 3-Axis Concept

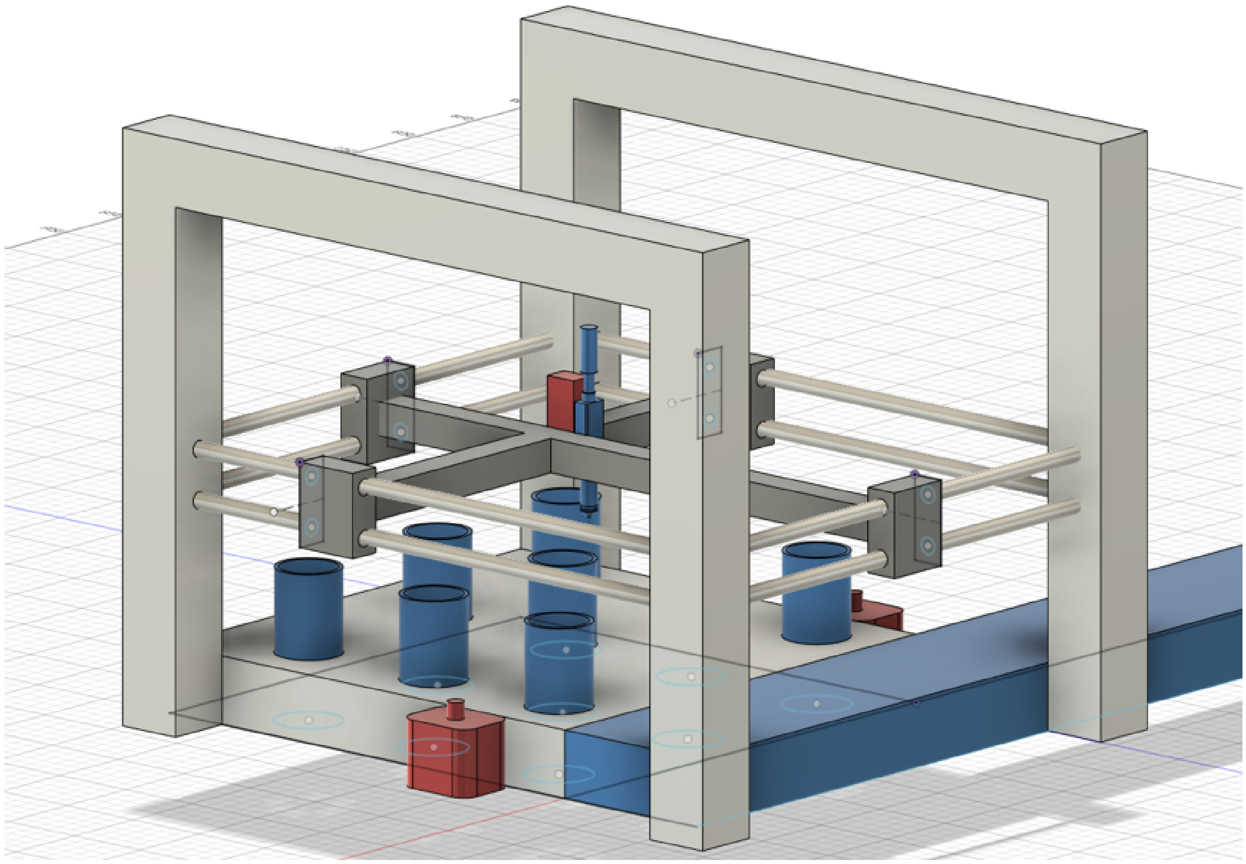


Figure 41: 3-Axis Concept

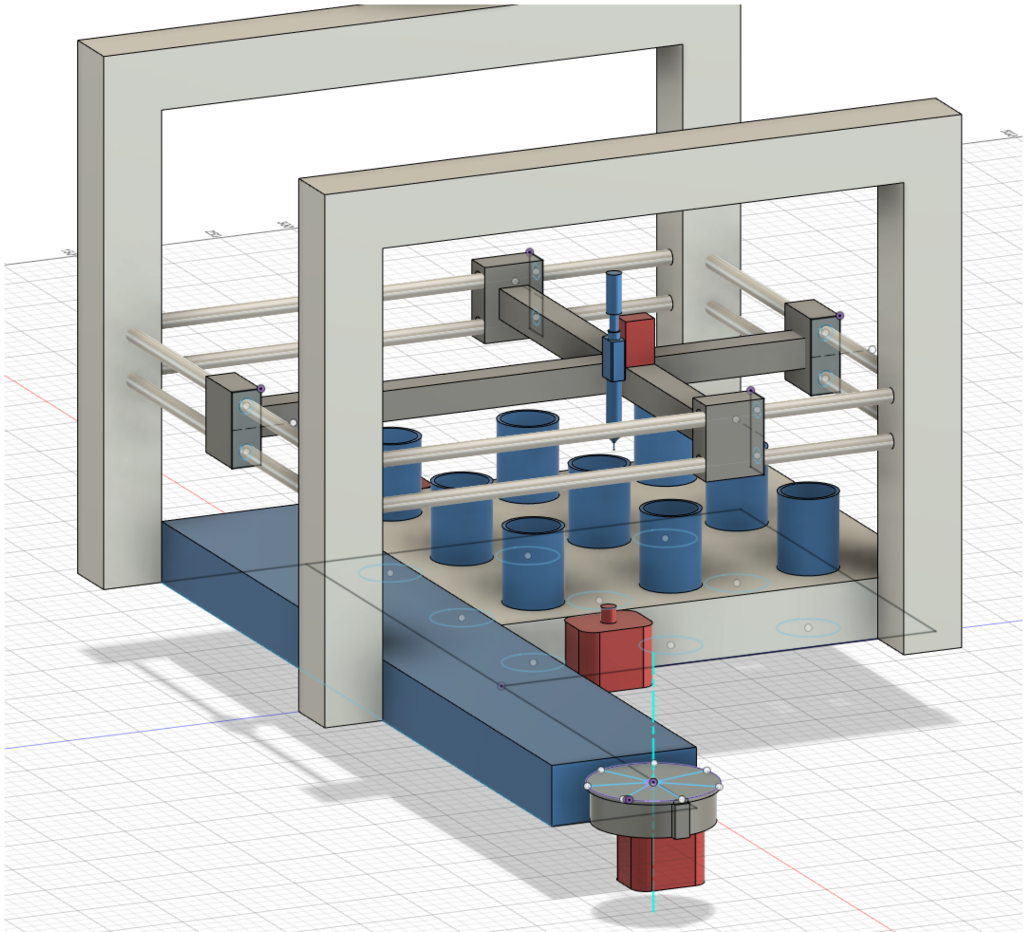


Figure 42: 3-Axis Concept

Concept Evaluation & Recommendation

In the previous three sections, three distinct concepts were presented to perform the desired functions and address the specific problem set of sorting medication. Of these three concepts, concept #1 and concept #2 are the most highly refined and simplistic. Concept #1 has the key advantage over #2 in that it is a simple single axis machine with servo controlled rotation of the sorted pills bin. The drawback with this design is that the machine is dependent on the medication being channeled into the bottom so that the head mechanism may pick up the medication and move it into the pill bin, so in the event of a jamming of the channel line the machine will seize. As for concept #2, there is no gantry style movement, only rotation and extension/retraction. This method of movement is ideal as the machine is working around a single pivot point at all time allowing for fast and efficient movements which will be imperative to the success of this machine. If the machine is not able to perform the task of sorting pills into bins faster than at least one pharmacy technician, then there is no reason for a business to consider adopting this technology. This is the major concern with recommending concept #1 as the amount of movement required for each pill cycle is quite high. From this evaluation alone concept #2 is seen to be the most promising.

Now that the three current concepts have been evaluated on their own the choice of concept #2 must be verified. This will be done through the use of a Pugh Matrix using the concept #2 as the datum as it is considered by the team as the most promising. Within the Pugh matrix the following evaluation criteria will be used: speed, safety, simplicity, scalability, efficiency, cost, and size. Each of these evaluation criteria have the following definitions for this matrix.

- **Speed** - Time taken to complete a pill retrieval cycle
- **Safety** - Possibility for pill mismanagement
    - ie. depositing medications into incorrect bins or cross-contamination
- **Simplicity** - Number of moving parts
- **Scalability** - Ability to add additional pills
- **Efficiency** - The efficiency in the machine movements
- **Cost** - Number of electronic components
- **Size** - Overall footprint or area/volume

Upon completion of the Pugh's matrix using these criteria the following is produced.

Table 11: Pugh Matrix

Evaluation	Concept #1	Concept #2	Concept #3
Speed	-	Datum	-
Safety	-		-
Simplicity	+		-
Scalability	+		-
Efficiency	-		-
Cost	+		+
Size	-		-
Total +	3		1
Total -	4		6
Total	-1		-5

From the Pugh Matrix using concept #2 as the datum it is seen that concept #1 and concept #3 evaluate below the datum. Looking more closely it is found that concept #1 is just below concept #2 while concept #3 is far below the other concepts. The leads to the following recommendation of concept #2 for further development.

Further Development

As was decided upon in the conceptual design section, the concept to be further developed was #2. In concept #2 a SMT pick and place head is positioned in the center of a platform surrounded by the pill bins where pills are to be retrieved from and placed into the correct slot in the sorted pills bin. In the initial concept, a single platform existed that held all of the electronics and pills. The major issue with structuring the system in this way is that the user may easily access the electronics and tamper with the system, which is undesirable. Additionally, if a pill were to be mismanaged the pill could be dropped into the electronics and it may be lost to the user which poses a safety risk. Another potential issue with this design leading into a potential prototype is that having multiple pill storage levels is not necessary and adds additional complexity that is not needed for the proof of concept.

To address these concerns the following changes have been made to the initial concept. First off, the entire system is structured into three tiered platforms of acrylic. The first platform is used to house all of the electronic components excluding the SMT head and linear actuator. The second platform is used to house the SMT head, linear actuator, pill bins, and sorted pill bins. Finally, the top most platform is strictly to act as a barrier between the user and the machine preventing any tampering or misuse during operation. This top most platform at this moment will function as a lid so that the user may gain access to the middle layer between operations. Additionally, by structuring the system this way if a pill drops onto the platform it will not be lost in the electronics and will be more obvious to the user that an error has occurred in operation. The other major change that has been made is that the second tier of pill bins has been removed, but some ideas for adding more pills revolve around making the pill bins a movable wall rather than moving the SMT unit vertically. Lastly, in the original design servo motors were the initial implementation but stepper motors would be better here as they have improved speed and the level of control of servos is not needed. For a visual representation of this configuration see all of the figures of the 3D model below.

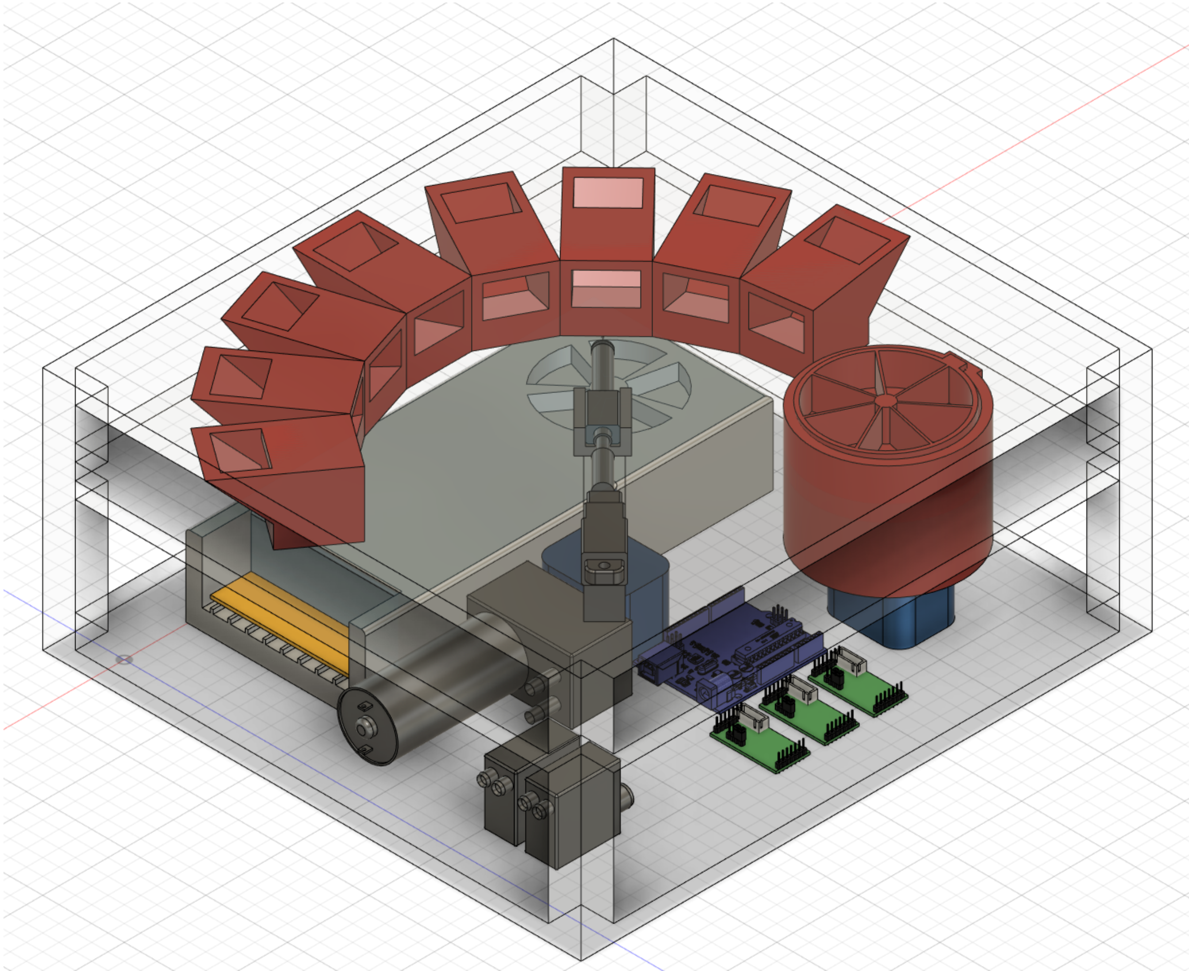


Figure 43: Top front-left view

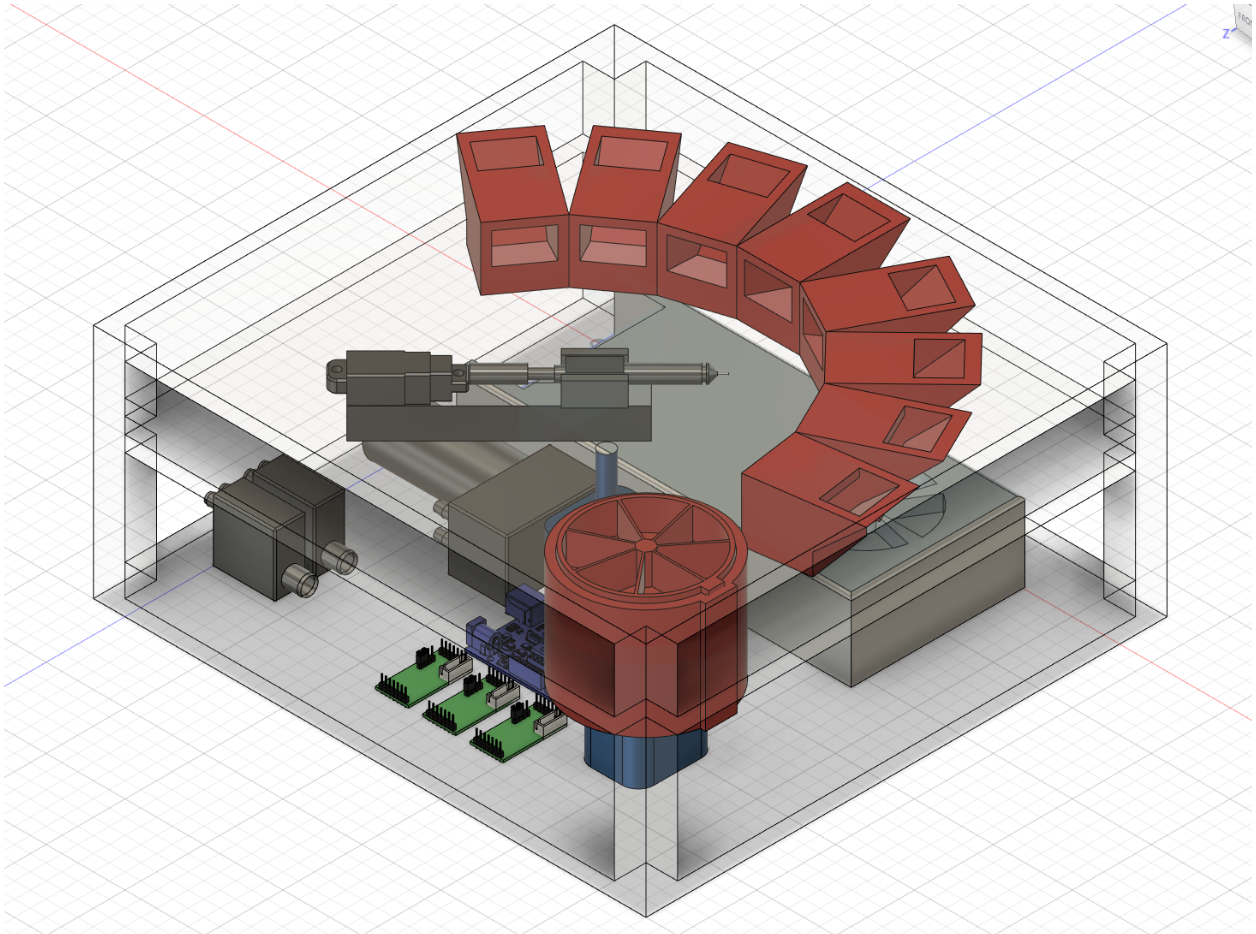


Figure 44: Top front-right view

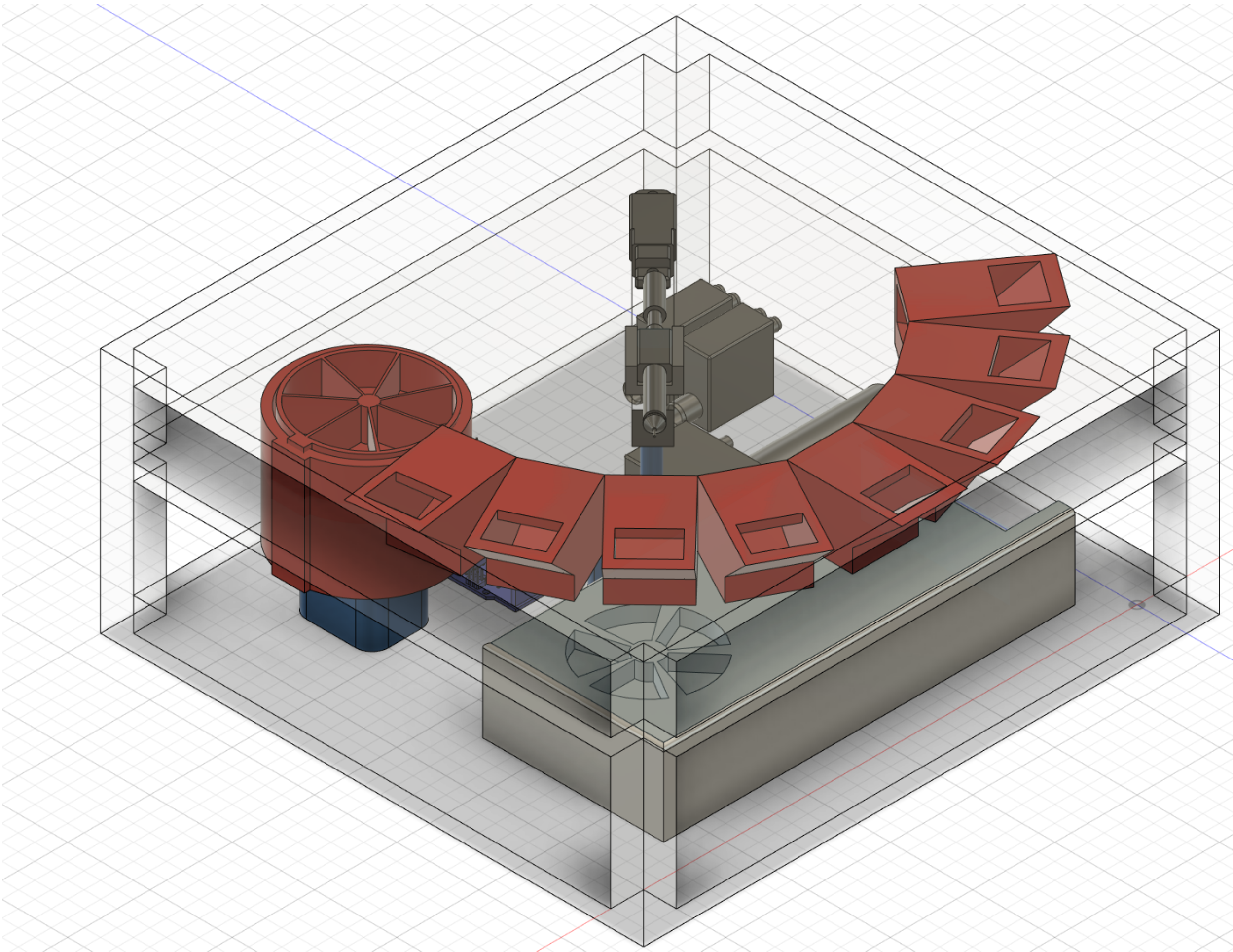


Figure 45: Top back-left view

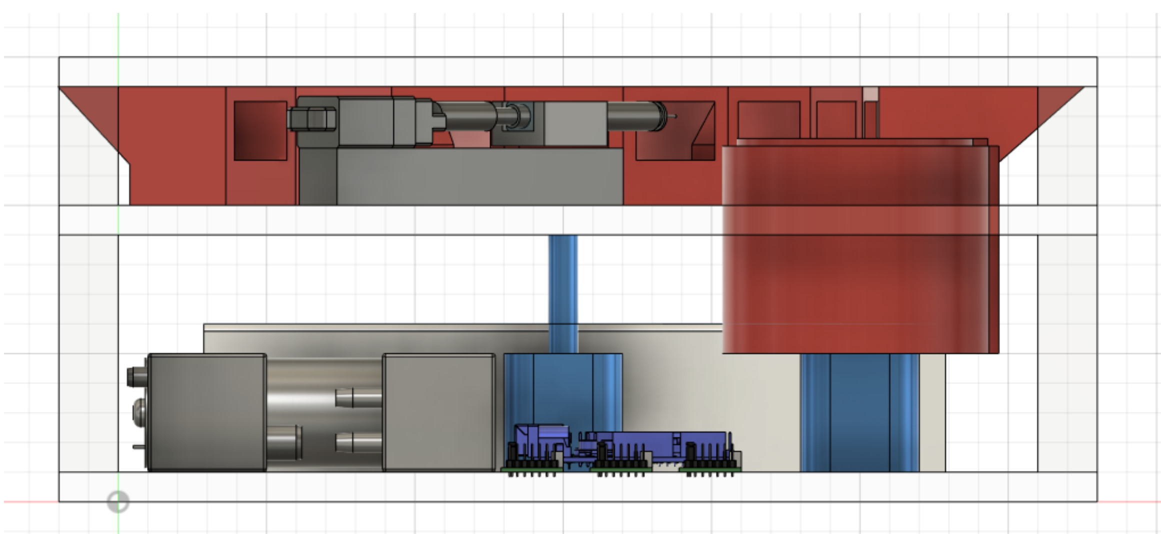


Figure 46: Front view

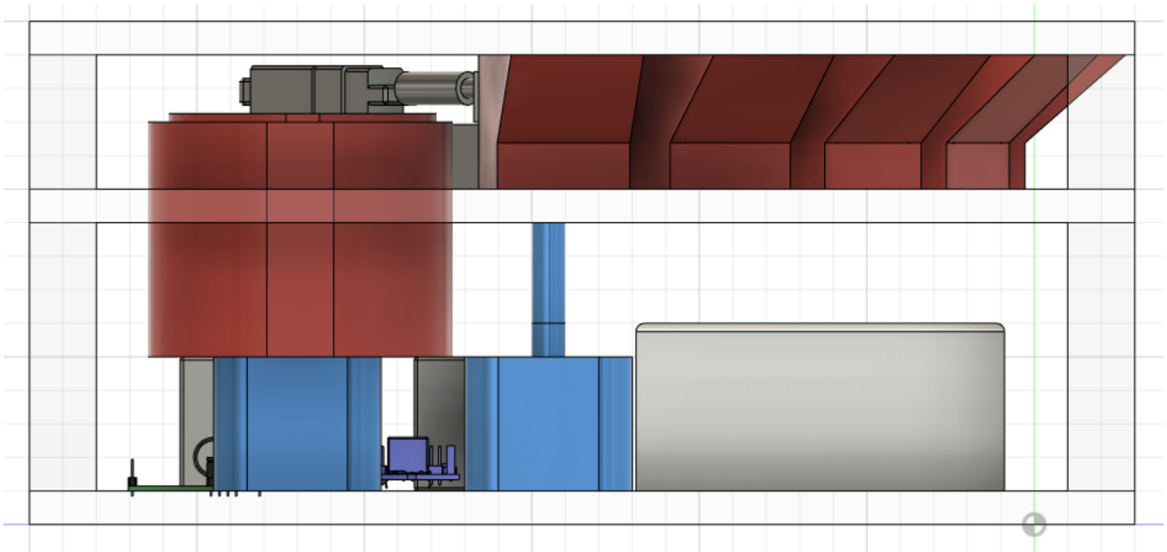


Figure 47: Right view

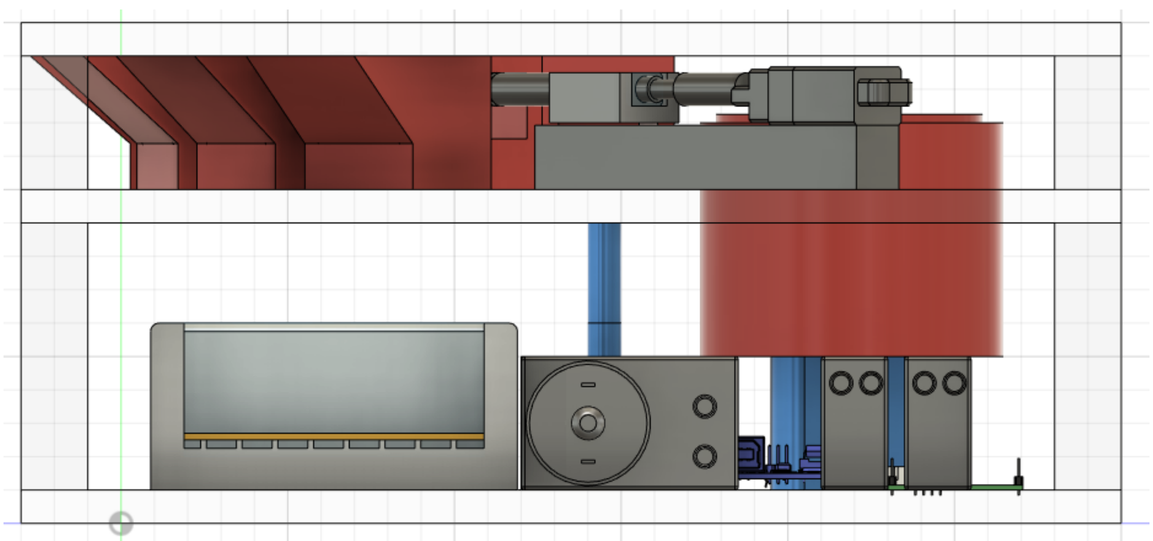


Figure 48: Left view

Now in addition to the changes noted above the major electronics for this implementation have been included in the CAD. The electronics pictured in this model have been tagged in the figure below. Of note in this figure the system will operate on a 110V AC power supply. In order to drive the SMT head and linear actuator a 12V vacuum pump will be used along with two solenoid valves. In order to drive the rotation of the SMT and the sorted pills bin two 12V stepper motors will be used. Finally, in order to control these electronics an arduino uno will be used along with stepper driver modules. One last thing that has been decided upon is that the pill bins and sorted pill bins with be prototyped in either PLA or ABS plastic with polypropylene being the plastic used for mass production as it will allow for the plastic to be recycled and better adhere to any issues the FDA or DEA has with pills being in contact with the bins.

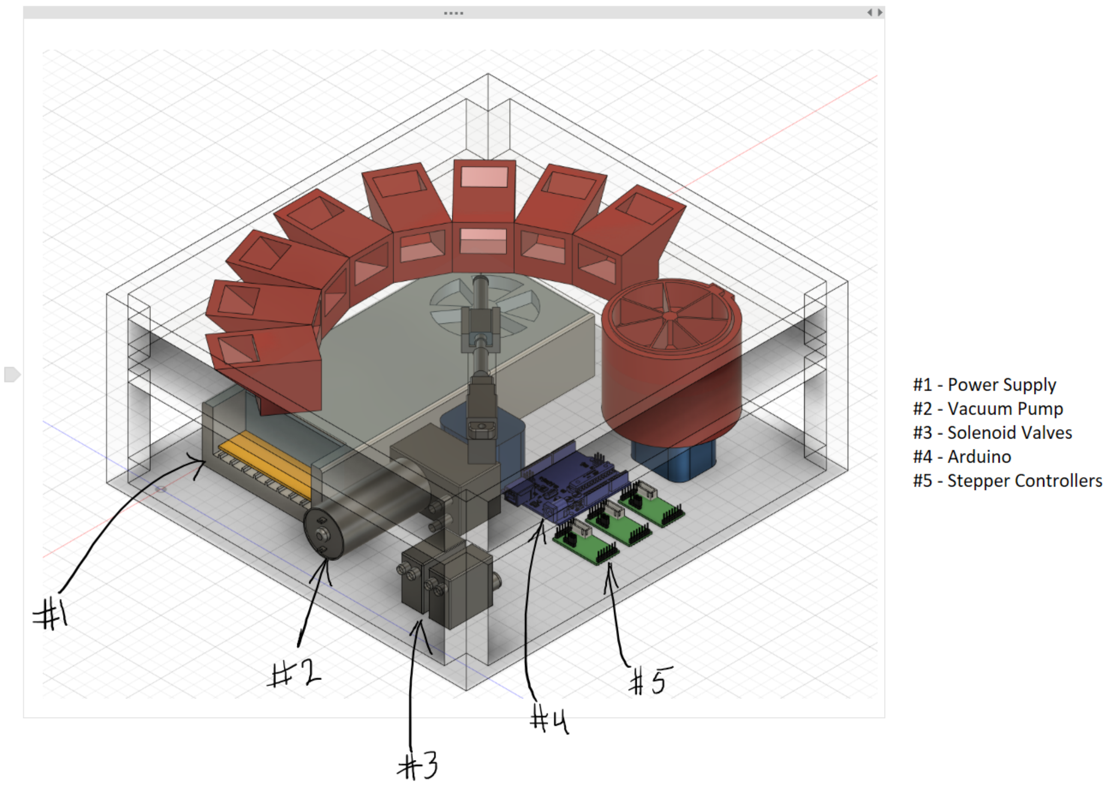


Figure 49: New components

As for the safety requirements of this project the change in structure of the system by using a tiered system will provide multiple improvements to safety by preventing the user from tampering with the machine during operation. Additionally, as was previously stated, making the end products pill bins made of polypropylene will allow for disposable pill bins to adhere to any issues with controlled substances stated by the FDA, DEA, or EPA. Finally, in terms of the value requirements the prototype is estimated to cost less than the initial requirement of $500 with a maximum price being $400 and the minimum being $265 depending on product availability due to the COVID-19 pandemic. A table of the averaged cost for the components can be found in the table below. As for the desired weekly operating cost of $1 this design optimizes movements and minimizes the required electrical components making the most of any electricity used. For the final requirement of having an intuitive operating system that will come down to the software design at later stages of development.

Table 12: Machine cost

Prototype Cost	
SMT Assembly ($)	180
Materials ($)	60
Power Supply ($)	20
Vacuum Pump ($)	20
Controls ($)	20
Stepper Motors ($)	40
Total ($)	340

The final topic of discussion for this design is how the user will interact with the system. In order to begin, first the user must scan a barcode that will load the necessary information for sorting into the software. From this point the system will indicate where pills are to be positioned in the bin locations. Once the pills have been loaded into their correct locations the user will put the “lid” back on top of the system allowing the system to begin sorting. As the system begins sorting the user may do other things, but in the event that there is an error during sorting the machine will notify the user through an alarm and the user will need to address the issue. Once all pills have been sorted into their appropriate locations, the sorted pill container may be removed and the lid placed on top. From this point, the process may occur again by the user repeating these steps.

IV. Embodiment Design

Final Concept Description

In developing the final design for prototyping, there were a few areas that needed further iteration and improvement. The largest change made is to the central sorting head unit which previously was simply an SMT head and linear actuator directly attached to a stepper motor via some bearing. First off, the head unit has been changed so that pills will now be retrieved overhead which is inspired by a SCARA robotic arm minus one axis. This change is primarily motivated by a need for the central head unit to reliably retrieve the medication from the bin which previously was thought to be inconsistent. Located below are the initial sketches of what this change would look like.

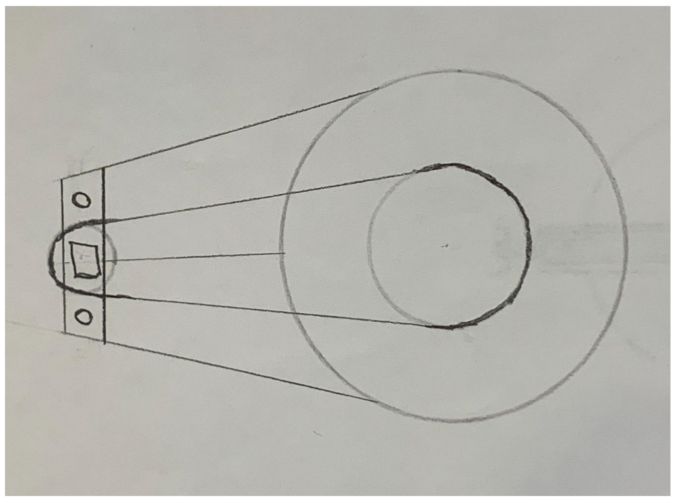


Figure 50: Central Head Unit - Top

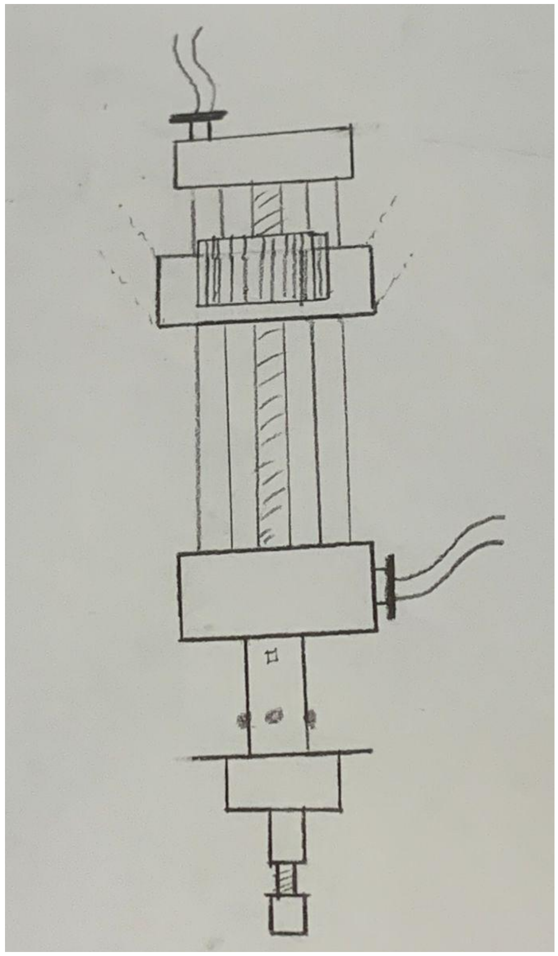


Figure 51: Central Head Unit - Picker Assembly

As can be seen in the sketch above the retrieval mechanism causing the SMT to descend will be driven indirectly via two meshing gears on a threaded rod. Alternatively, this could be accomplished via a belt allowing for a gear ratio to improve speeds. To achieve the rotation of the entire head unit, it will be mounted to a large bearing that meshes with a gear driven by a stepper motor. As it was in the previous design, grabbing of the medication will still be done with an SMT head powered by a 12V vacuum pump and a pneumatic solenoid valve. Located in the figures below are the 3D models of the redesigned central unit and its individual parts.

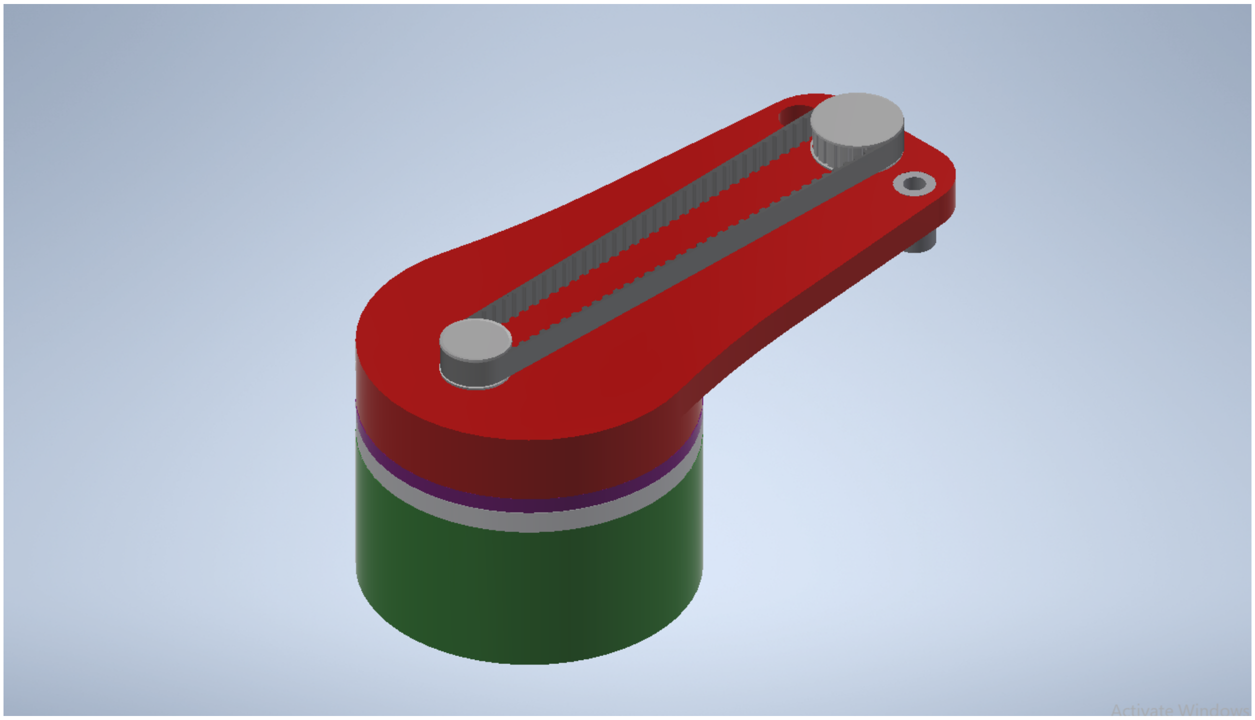


Figure 52: Assembled Head Unit

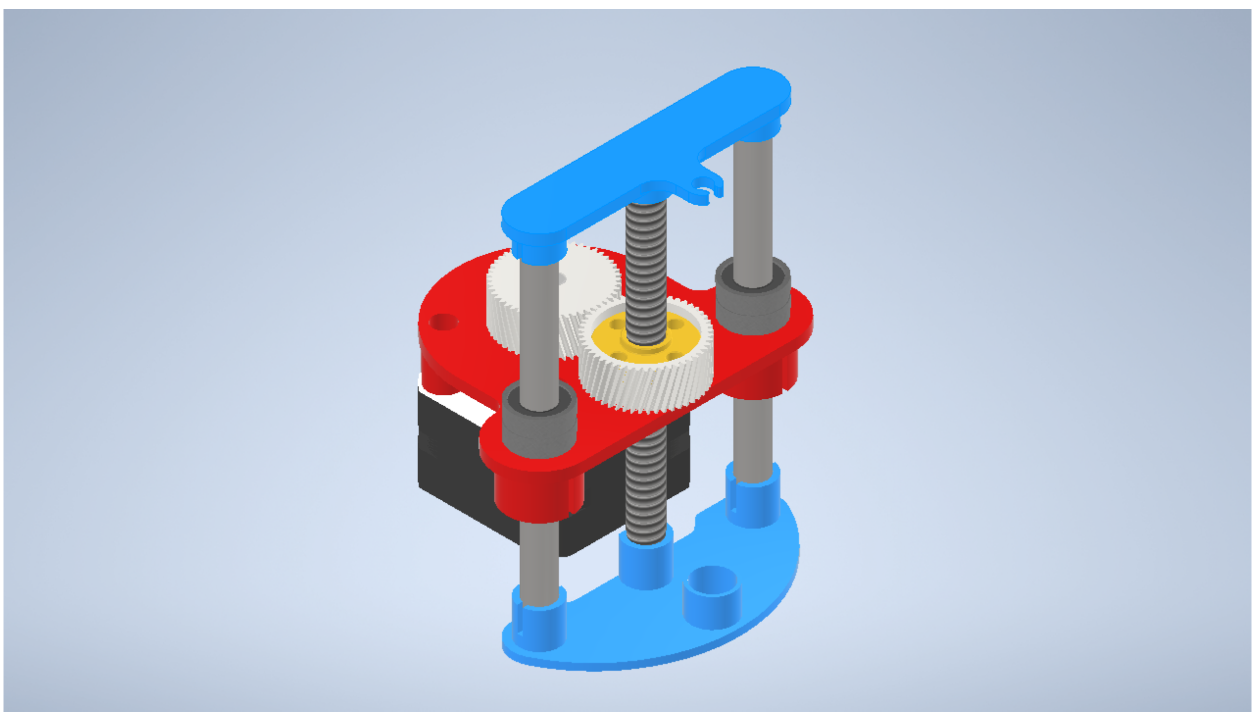


Figure 53: Head Unit Z-axis w/o Belt

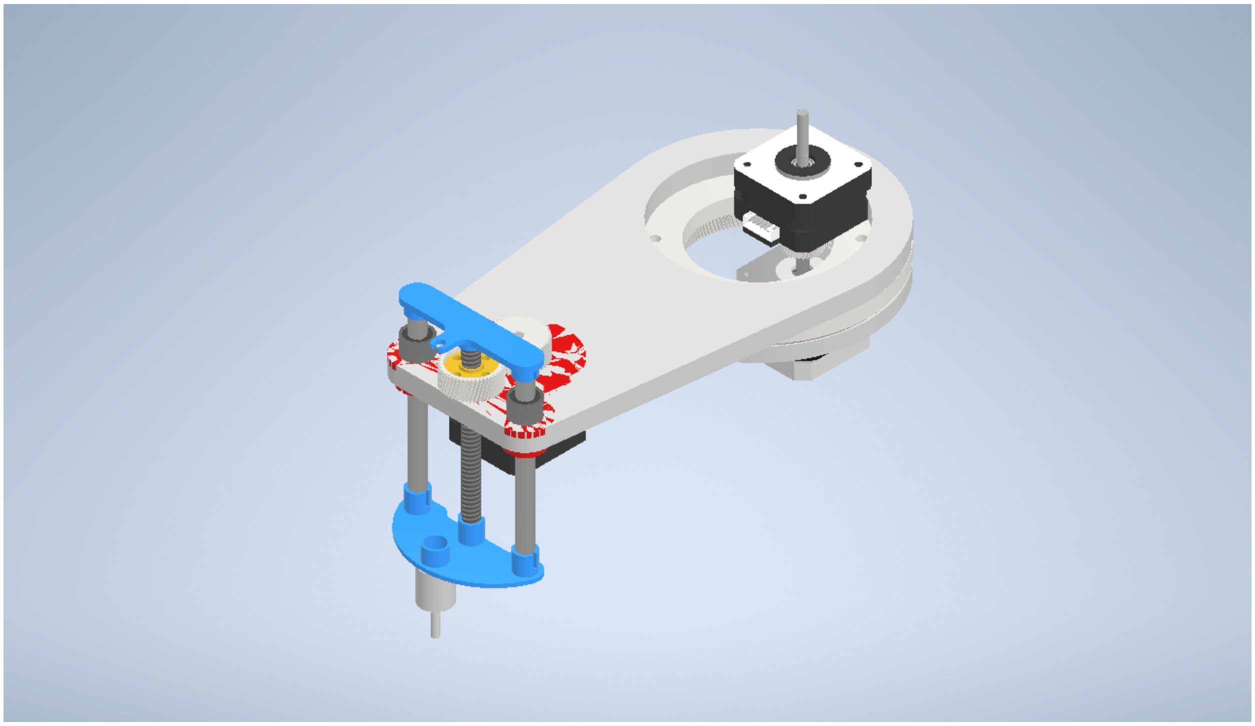


Figure 54: Head Unit Arm Assembly

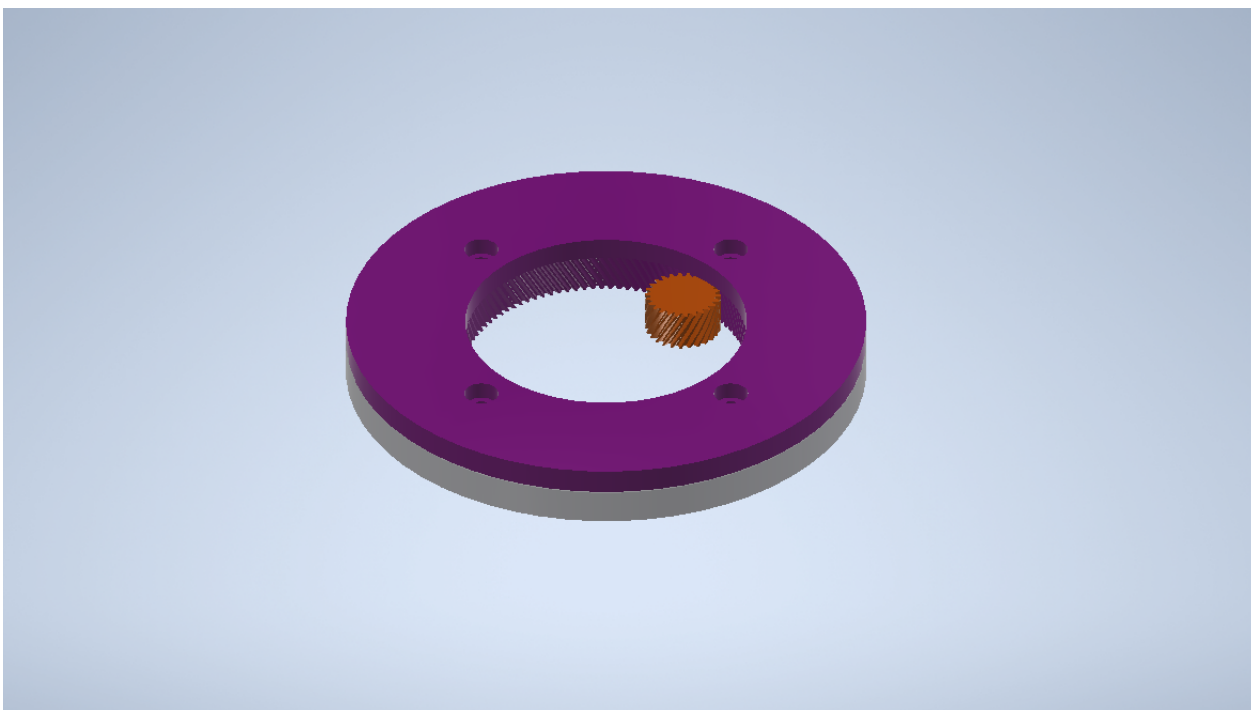


Figure 55: Head Unit Rotation

Now, since pills will be retrieved from above, the containers must also be modified. In modifying the pill bins the problem of the user loading the medications has also been addressed by allowing for pills to be loaded via funnels. The new pill bin models can be seen in the figures below as well as an assembly using the funnel idea. As can be seen, the new pill bins allow for medication to be retrieved from the top and filled from the rear. Additionally, they are designed to lock into place on the assembled unit via a sliding snap closure on the bottom.

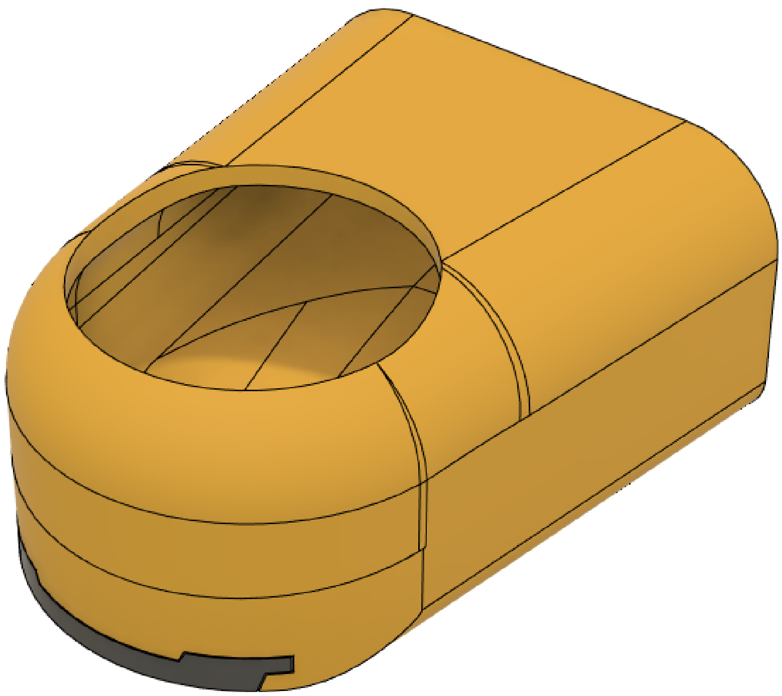


Figure 56: Pill Storage Bin

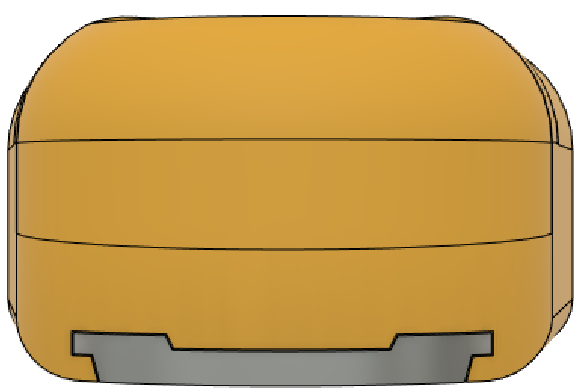


Figure 57: Pill Storage Bin

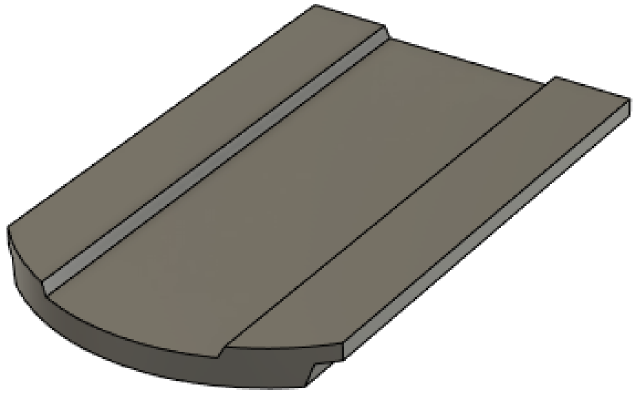


Figure 58: Pill Storage Bin - Sliding Snap Closure

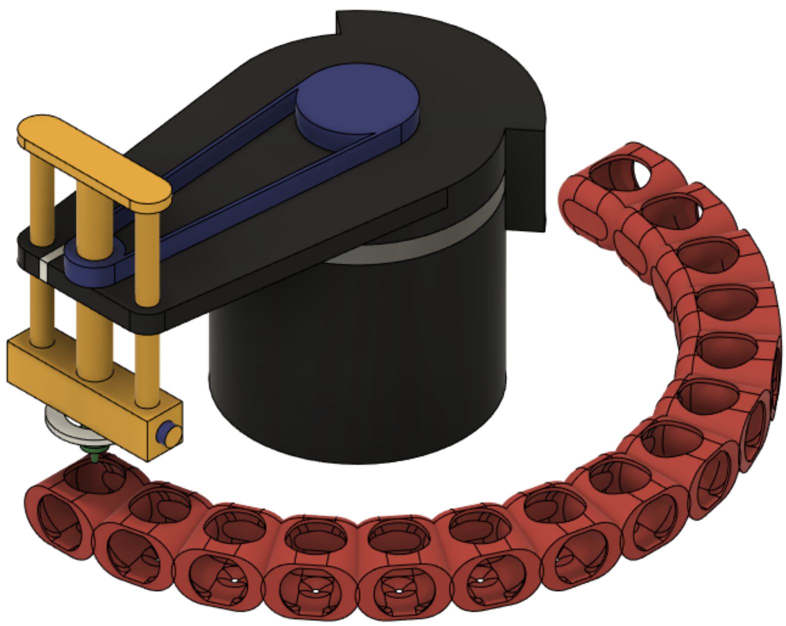


Figure 59: Basic Assembly

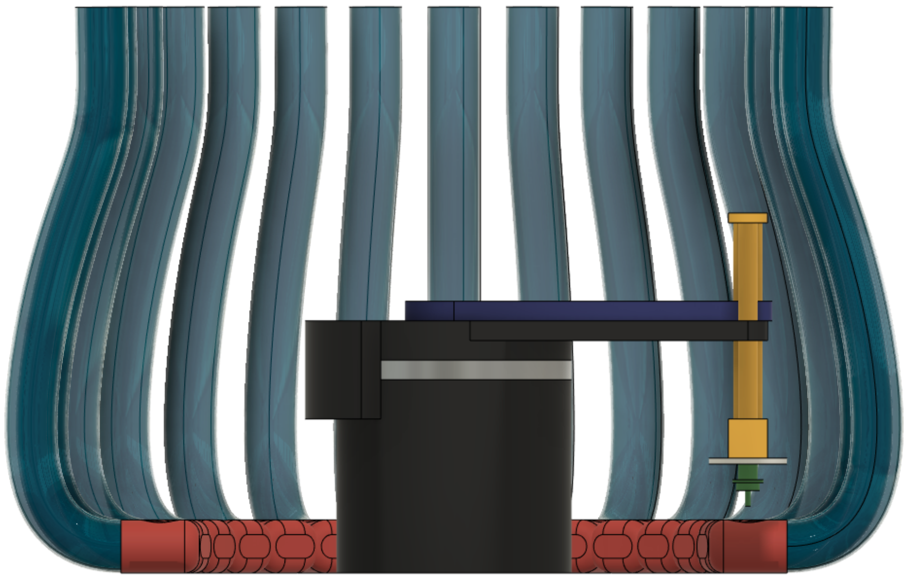


Figure 60: Basic Assembly w/ Funnels

Next, there was a desire to change how the sorted pills bin mounts to the assembled unit while also allowing for the use of RFID if desired. The main idea with this change was to allow for the sorted bin to only be inserted one way and to be easily placed and removed. This was accomplished by using magnets and placing the RFID sensor on the base plate. Figures of this design can be seen below.

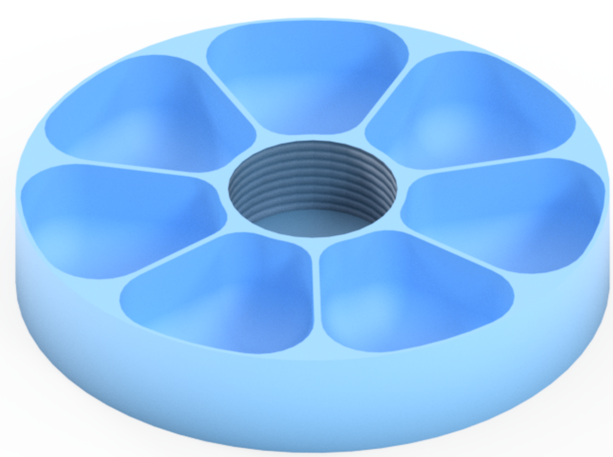


Figure 61: Sorted Pill Bin

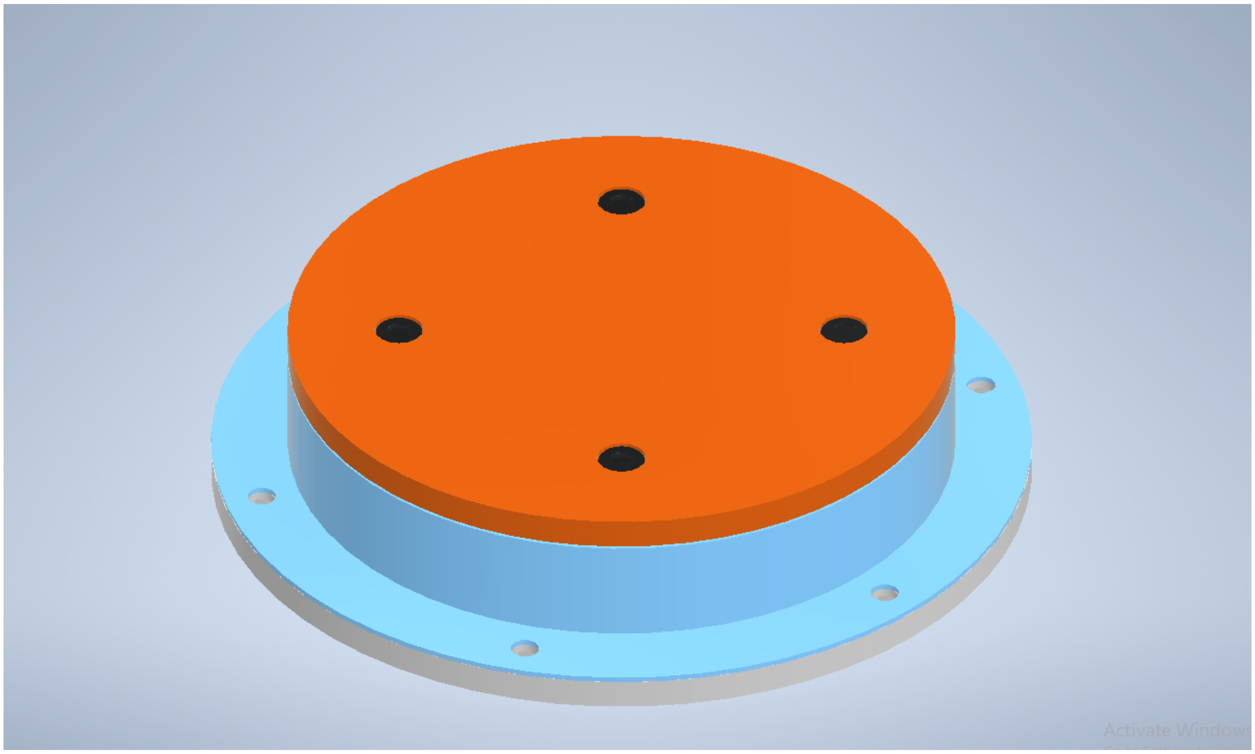


Figure 62: Sorted Pill Bin Base Plate

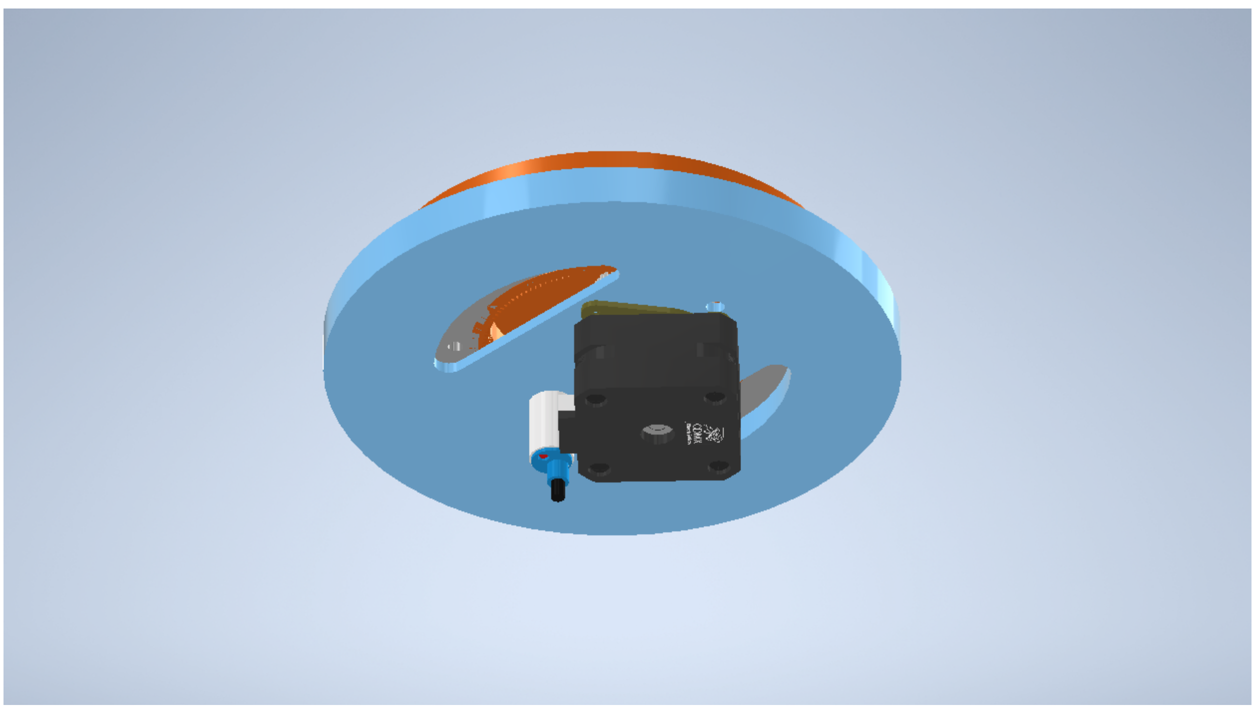


Figure 63: Sorted Pill Bin Base Plate

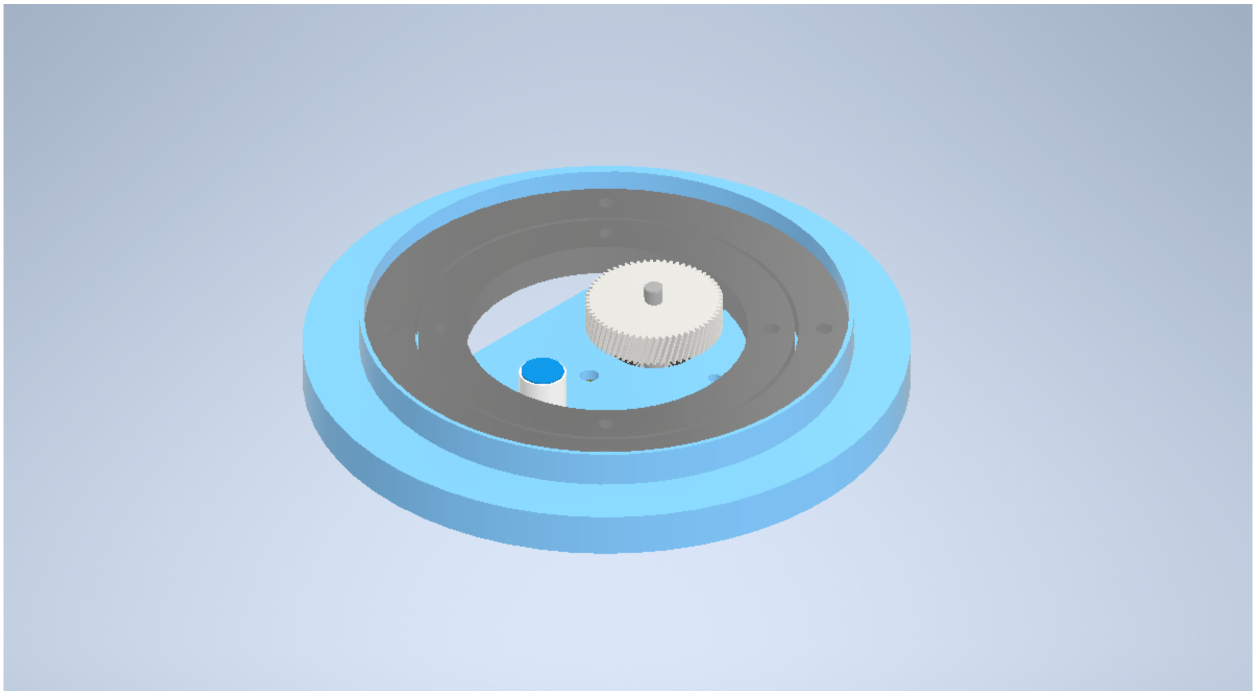


Figure 64: Sorted Pill Bin Base Plate

Those are the major changes to the previous design. The notable new additions to the design are some leds around the pill bins that act as an indicator to the user on the machine's state and a protection panel for the power supply. Lastly, there are some final design considerations for the prototype when it comes to mounting all of the components. For prototyping, the mounting of the components will be done primarily through screws and glue if necessary. Located below are some figures showing how the various parts shall be mounted and the new power supply closure.

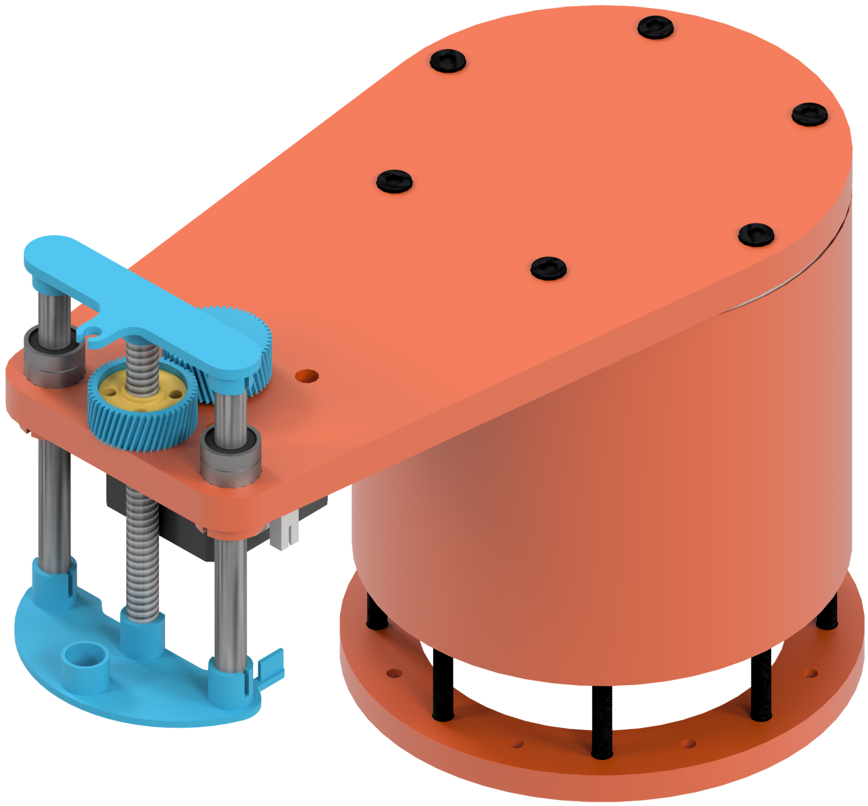


Figure 65: Head Unit Mounting

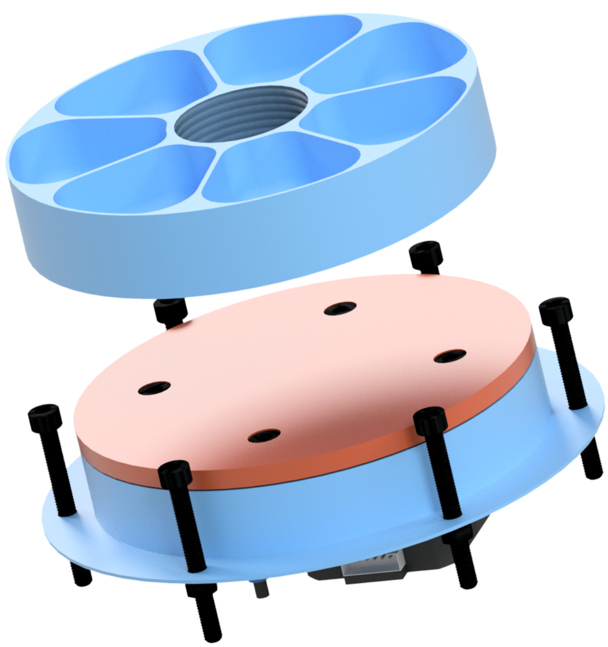


Figure 66: Sorted Pill Bin Base Plate Mounting

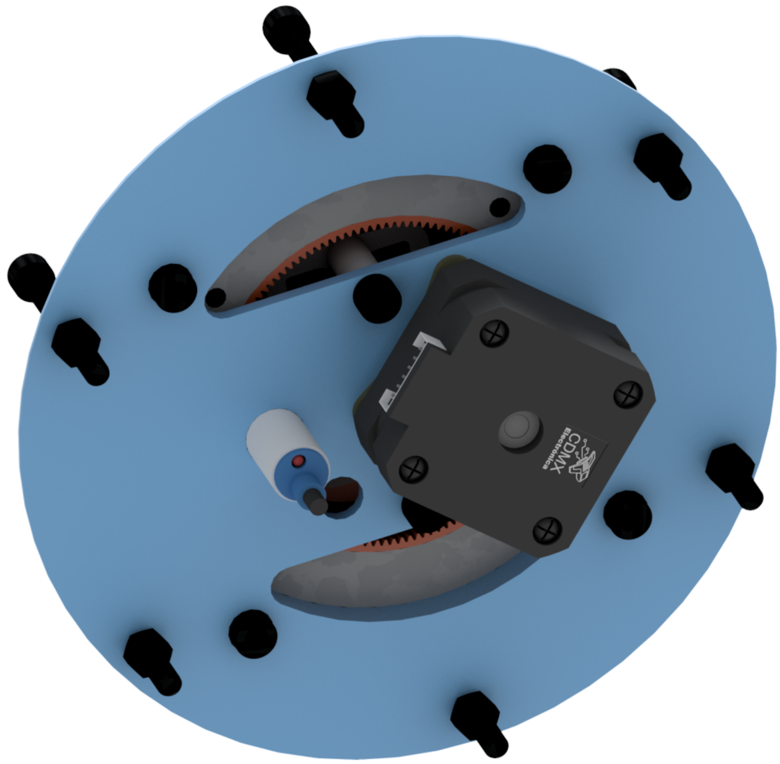


Figure 67: Sorted Pill Bin Base Plate Mounting

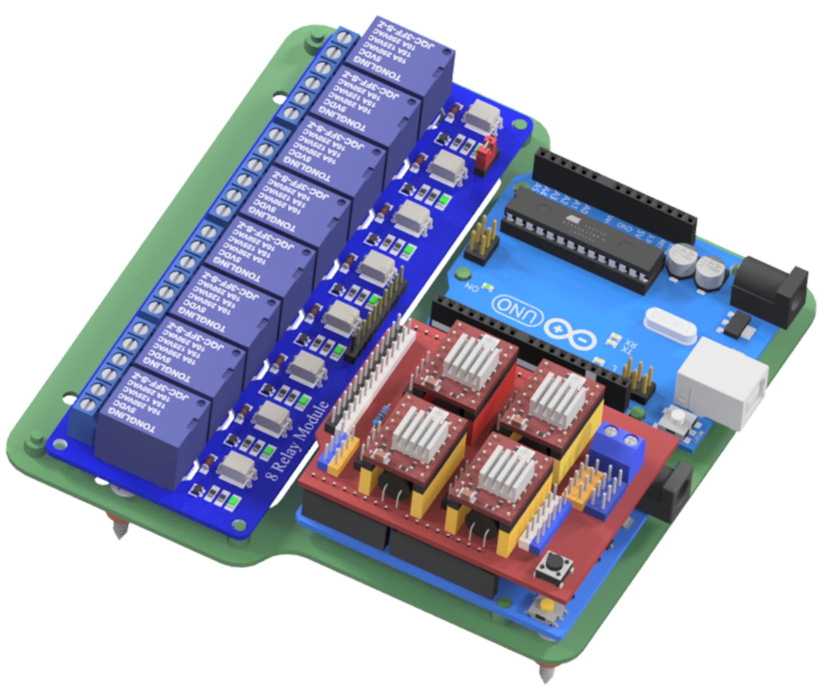


Figure 68: Electronics Mounting

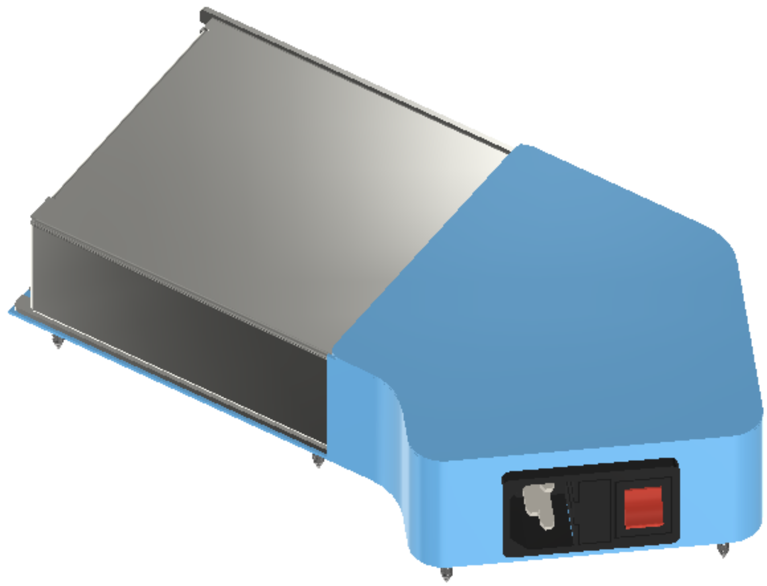


Figure 69: Power Supply Closure

Finally, the design has shifted from a 3-tiered design to a 2-tier design with the idea to add some type of cloche later. For the base of the entire assembly a circular wood panel is cut using a CNC to create all the mounting points and access for electronics and pneumatic tubing. Additionally, to elevate the board feet have been designed so that the electronics may be mounted below. The final concept's 3D model can be seen in the figures below.

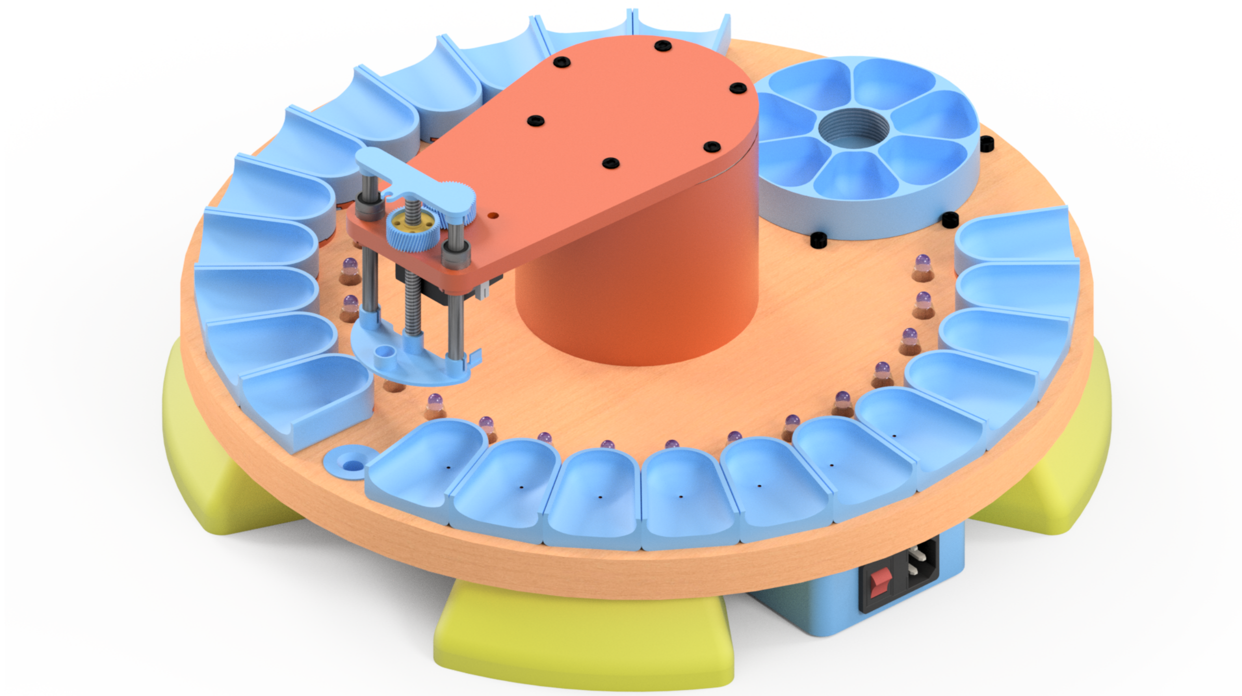


Figure 70: Final Construction

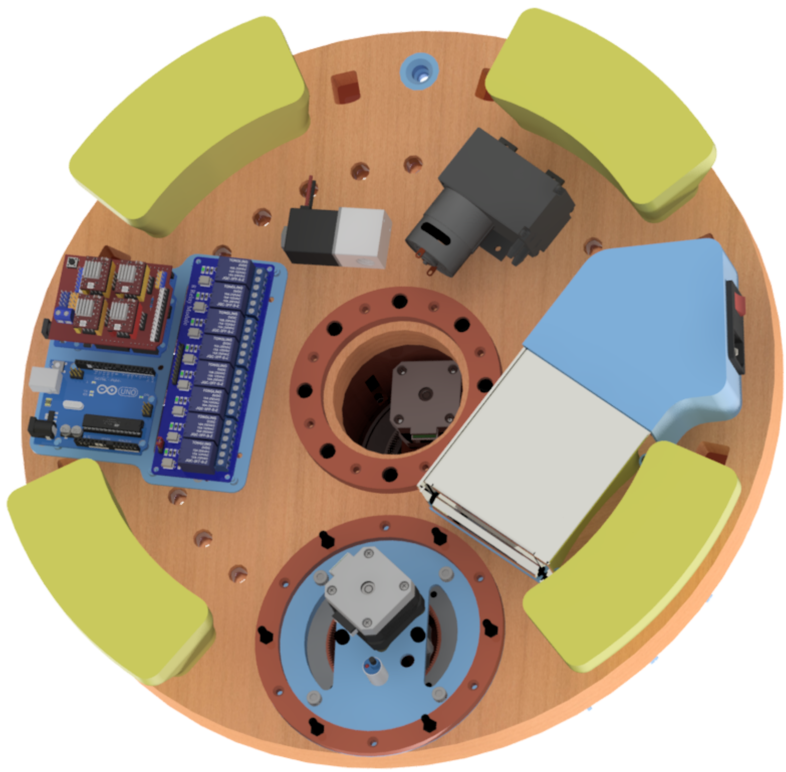


Figure 71: Final Construction - Bottom

Sizing

To determine the sizing of the electronics noted previously and in the labeled CAD model above, the following calculations were performed. The first piece of equipment that was to be sized was the stepper motor responsible for rotating the central head unit. The most important consideration for this piece of equipment is the load to rotor inertia ratio so that the stepper does not drift when trying to stop. To obtain this number first the load inertia must be calculated which in this case has the shaft located away from the center of mass of the unit. Performing this calculation results in an inertia value of 22215.0 g-cm^2. This value seems too high for what this project is trying to achieve so it might be best to move the stepper motors shaft to the center of mass of the unit while also reducing the weight of the central unit through different material selection. In doing so if the density of the unit could be reduced to ~2 g/cm^3 the inertia value could be reduced to 6651.5525 g-cm^2. Using both of these values the required rotor inertia can be calculated. In an ideal scenario a 1:1 ratio is desired but up to a 10:1 will function without error. This results in inertia ratio values for the 1:1, 5:1, and 10:1 cases of 6651.5525, 1330.0, and 665.0 respectively. This leads to the need for either a nema 17 or 23 stepper motor with 23 being more favorable as it will provide a better inertia ratio given that nema 23 are often rated at a higher rotor inertia. A summary of the calculations can be found in the table below.

Table 13: SMT Unit Stepper Sizing

SMT Unit Stepper Motor	
SMT Inertia - off-axis (g-cm2)	22215
SMT inertia - centered (g-cm2)	6651.5525
Rotor inertia "10:1"	665
Rotor inertia "5:1"	1330
Rotor inertia "1:1"	6651.553

The next component to be sized is the stepper motor for the sorted pill bin. For this bin it will be assumed that it is fully loaded with pills and is a solid disk rotating on its central axis. The calculations for this component come out to be very similar to the central units stepper motor and are summarized in the table below.

Table 14: Sorted Pills Stepper Sizing

Sorted Pills Stepper Motor	
Sorted pills inertia (g-cm2)	6550.713
Rotor inertia "10:1"	655
Rotor inertia "5:1"	1310
Rotor inertia "1:1"	6550.713

The next component for sizing is the vacuum pump. For this component the main consideration is that the vacuum pump is capable of driving the solenoid valves controlling the linear actuator and SMT head. For these valves they are rated at a required operating pressure of 0.02 MPa - 0.8 MPa which allows for a lot of choice. Using these numbers it is known that the vacuum pump that is chosen must have a maximum positive pressure of at least 0.02 MPa. Upon looking at 12V vacuum pumps available on the market most have a maximum positive pressure of ~800 mmHg which equates to ~0.1 MPa.

The final piece that must be sized is the 110V AC power supply. To find the wattage required of the power supply the individual components maximum power was totaled. In doing so the total maximum wattage is expected to be ~90 W without any control boards. This means that a 120 W power supply would be sufficient. The summary of the power calculations can be found in the table below.

Table 15: Power Supply Sizing

Power Supply					
	Voltage (V)	Current(A)		Power (W)	
		Min	Max	Min	Max
Stepper Motor	12	0.5	1	6	12
SMT Head	5	0.6	0.8	3	4
Linear Actuator	12	1.667	2.5	20	30
Vacuum Pump	12	0.3	0.6	4	6
Solenoid Valve	12	0.5	1	6	12
Total				51	88

With this design all of the design requirements listed in the need analysis section will be satisfied. In terms of the performance requirements the design will be able to accommodate all pill sizes and textures due to the pill bin sizing and use of an SMT head. The only performance requirements that will require some further development are hitting the desired 99.99% accuracy and the desired fill rate of 60 prescriptions per hour. Obtaining the desired accuracy will be primarily accomplished through software while the speed will be dependent on the stepper motors selected. As for the size requirements the machine will function on 110V AC power and will fit well within the 600mm x 600mm x 600mm space. In terms of the mass of the entire machine it is estimated to be well under 20kg at ~5200g without pills, wires, pneumatic tubing, or mounting hardware. A table of the estimated weights of the components can be found below.

Table 16: Estimated Prototype Weight

Estimated Prototype Weight	
Wood (g)	1700
Central unit (g)	900
Power supply (g)	300
Sorted Pills (g)	1450
Vacuum Pump (g)	350
Solenoid Valves (g)	100
Controls (g)	100
Stepper motors (g)	300
Total (g)	5200

The entire assembly is designed to be built on an 18 inch circular board. With a depth of 24.5 mm, there is ample room to attach components to both surfaces. Additionally the total apparatus is only 227 mm fitting well within the 600 mm requirement.

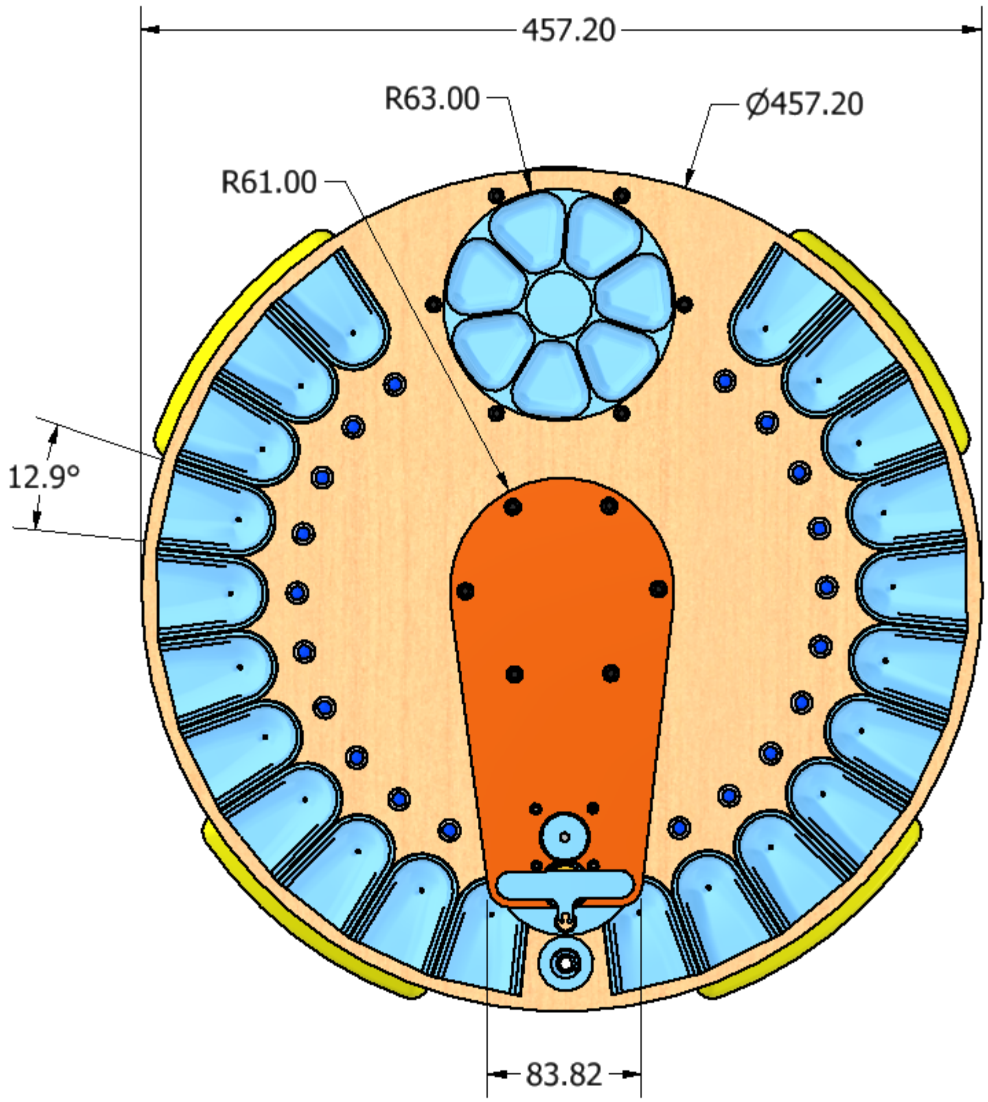


Figure 72: Top Profile Dimensions

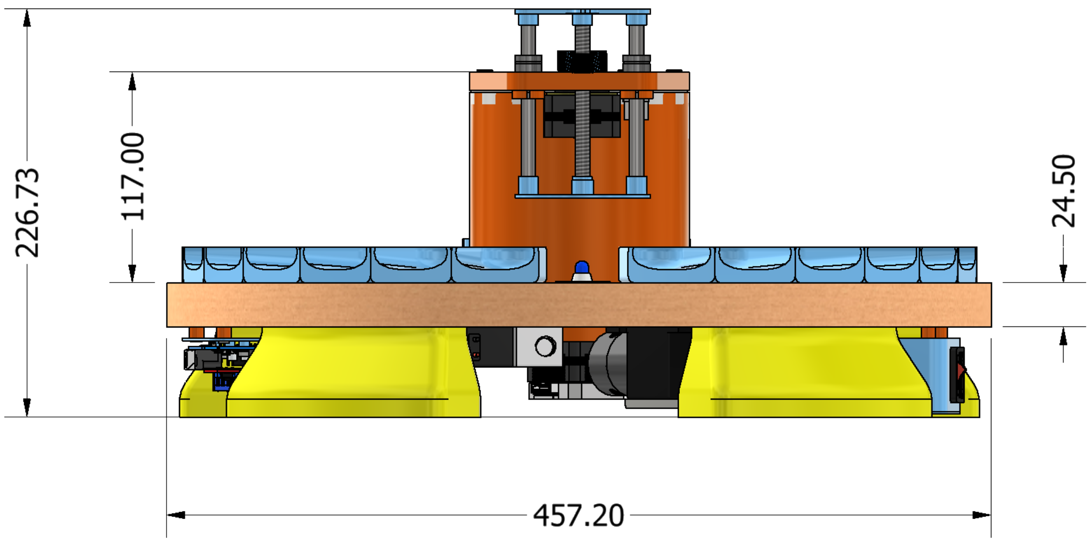


Figure 73: Side Profile Dimensions

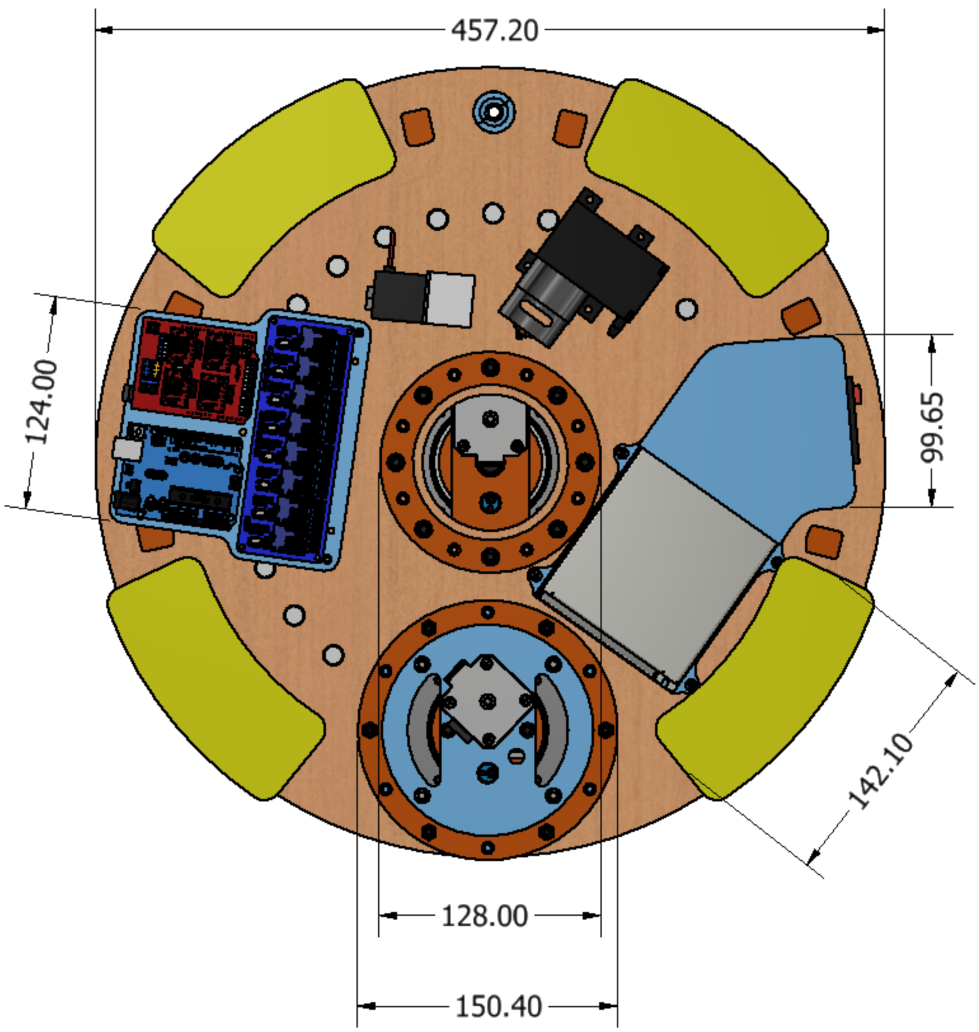


Figure 74: Bottom Profile Dimensions

Failure Mode Effects Analysis

Now given that this machine will be handling medications and thus requiring a high level of precision and accuracy, it is imperative that all potential modes of failure are identified. Additionally, it should be stated that for this project failure would be considered as any instance that causes a loss of precision/accuracy leading to a single pill being sorted incorrectly. To begin, the process of identifying potential modes of failure was done with the aid of an Ishikawa chart. The chart for this project can be seen in the figure below.

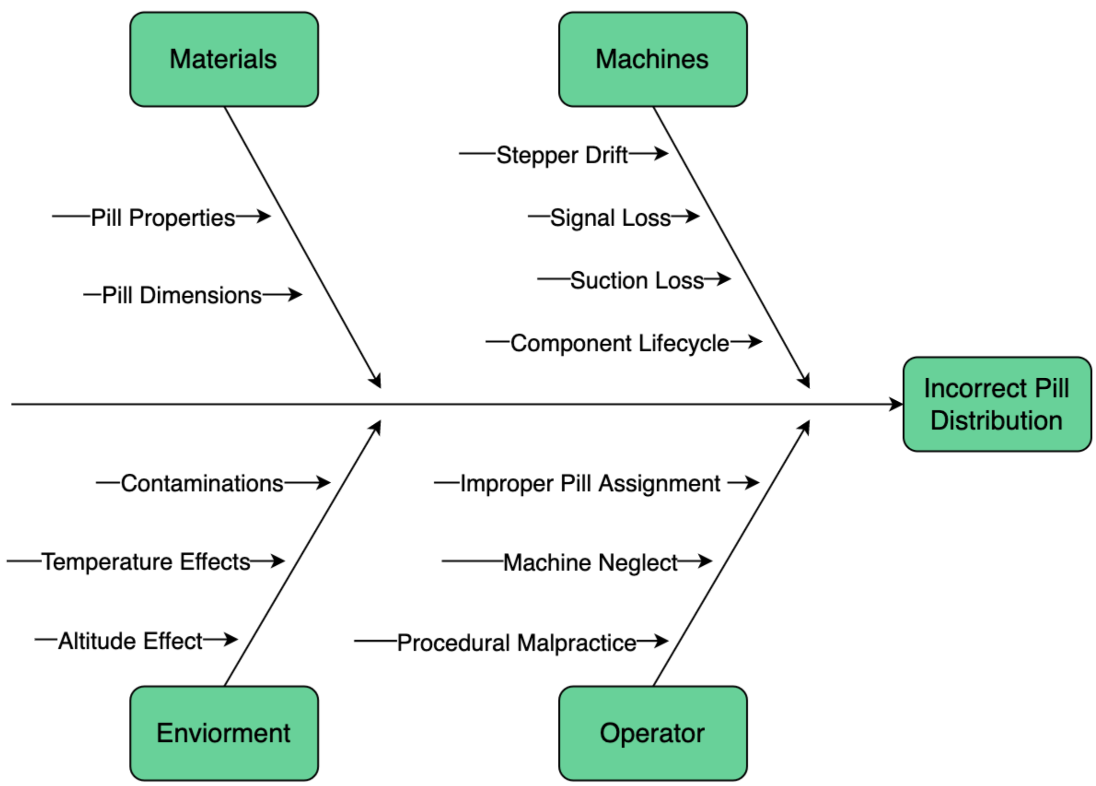


Figure 75: Ishikawa Chart

At the top and bottom of the chart located in the green rounded rectangles are the major categories of failure that lead to the incorrect pill distribution which is located on the right. From each of these major categories is a diagonal line with causes listed next to the diagonal line.

Starting with the materials category there are two main causes that were identified as potential causes of failure, both dealing with the pills themself. Given the large variety of pill shapes, sizes, and dimensions odd interactions will be likely. Additionally, not all pills will have the same texture, hardness, or weight. One pill could be a capsule which has a lot of give when compressed, while a tablet could have a powdery texture and be quite fragile.

Next is the machines category, which includes all of the electronics such as stepper motors, vacuum pump, and power supply. First, there is stepper drift which is a common problem that occurs over time which leads to a tuning requirement. To accompany this issue, it is not uncommon for stepper motors to experience a signal loss and move improperly. Next, there is the potential for suction loss which could occur for a multitude of reasons. Finally, there are component lifecycle concerns which become a greater concern the longer the machine will be in service.

There are also environmental concerns that need to be identified. Two major environmental concerns that are of concern are temperature and altitude effects. The primary concern with these two effects is they can lead to changes in other major components such as pill properties and loss of suction. To go along with these causes is contamination which follows the same logic as changes in temperature and pressure. For example, a small bit of dust could lead to a loss of suction.

The final major category has to do with the operator of the machine. There are three potential mistakes that the operator could make leading to incorrect pill distribution. First off, there is procedural malpractice which would be the operator simply not following the required procedure. Second, there is machine neglect, which could be a lack of tuning or cleaning by the user. Lastly, there is improper pill assignment which would look like the operator loading pills into the incorrect bin locations.

Assembly & Part Details

The assembly of this project requires many components beyond the electronic components that have already been mentioned. Located in the table below is a table of the required components along with their respective price at the time of order. The SMT head component is the most expensive on the list as it comes with an additional servo and a venturi vacuum system. Located below the total parts table are tables further breaking down what parts are used in each assembly of the device.

Table 17: Parts List

ORDER 1 - Submitted						
Name	Status	Catagories	Quantity	Price	Shipping	Est. Delivery
Power Supply	Completed	China/12v/120w	1	$12.46	Free	3/12
12 Vacuum Pump	Completed	NULL	1	$16.46	Free	03/03
CNC V3 Shield	Completed	NULL	1	$10.84	Free	3/11
SMT Head	Completed	BT1040ZSGA	1	$133.00	Free	03/06
Vacuum Tube 4mm	Completed	China/Blue/6x4mm-10m	1	$6.43	Free	03/12
Vacuum Tube 6.5mm	Completed	Blue/10x6.5mm-3m	1	$5.96	$0.91	03/12
Emergency Stop	Completed	NULL	1	$1.12	$1.05	03/28
Fused Power Socket	Completed	Red	1	$0.92	Included	3/28
Pnumatic Connector	Completed	PY/6mm	1	$0.51	Free	3/28
Juki Nozzle Holder	Completed	NULL	1	$10.50	Free	03/06
ORDER 2 - Submitted?						
Timing Belt	Completed	92 Teeth, 460mm Length	1	$10.79	Free	3/4
Pnumatic Tubing	Completed	4mm OD/Blue	1	$10.99	Free	3/3
Turntable Bearing	Completed	NULL	1	$10.99	Free	3/3
Relay Module	Completed	NULL	1	$10.99	Free	03/03
ORDER 3 - Submitted						
Lead Screw	Completed	NULL	1	$9.99	Free	3/8
Linear Rod (8mm)	Completed	8mm x 100mm	1	$9.99	Free	3/8
Linear Bearing (8mm)	Completed	Null	1	$10.95	Free	3/8
ORDER 4 - Submitted						
Slip Ring	Completed	NULL	1	$14.99	Free	3/27
Bearings	Completed	15x21x4	1	$9.99	Free	3/27
Turntable Bearing	Completed	NULL	1	$10.99	Free	3/27
Solenoid Valve	Completed	NULL	1	$9.45	Free	3/27
ORDER 5 - Submitted						
Nema 17 Damper	Completed	NULL	1	$10.99	Free	3/30
M5x14mm Hex Bolt	Completed	50 pcs	1	$9.99	Free	3/30
M5 Hex Nut	Completed	NULL	1	$7.62	Free	3/30
Hall Effect Sensors	Completed	NULL	1	$11.52	Free	3/30
ORDER 6 - Submitted						
Stepper Motor (13Ncm)	Completed	17HS4023(13Ncm) / 1PCS	2	$19.98	Free	4/7
Stepper Motor (42Ncm)	Completed	17HS4401S(42N.cm) / 1 PC	1	$10.99	Free	4/7
Current Sensor	Completed	NULL	1	$7.99	Free	4/8
Addressable Led	Completed	NULL	1	$13.99	Free	4/7
ORDER 7 - Submitted						
M5 Tap & Bit	Completed	M5x.8 / M5x.8 Tap - 4.20mm	1	$7.01	Free	4/16
RFID - RDM6300	Completed	NULL	1	$7.79	Free	4/16
RFID - Tags	Completed	NULL	1	$8.99	Free	4/16
M5 x 45mm Hex Bolt	Completed	M5 x 45mm / 50 pcs	1	$12.99	Free	4/17
Magnets	Completed	NULL	1	$12.99	Free	4/19
Wood Screws	Completed	#6 x 3/4"	1	$9.19	Free	4/17
Turntable Bearing	Completed	NULL	1	$10.99	Free	4/16
TOTALS			37	$481.33	$1.96	

Table 18: Sorting Arm Parts

Sorting Arm	
Item	Quantity
Juki Nozzle Holder	1
Timing Belt	1
Turntable Bearing	1
Lead Screw	1
Linear Rod	2
Linear Bearing	1
Slip Ring	1
Roller Bearing	1
M5 x 14 mm Hex Bolt	6

M5 Hex Nut	6
M5 x 45 mm Hex Bolt	6
Hall Effect Sensor	1
Stepper Motor 13Ncm	1
Stepper Motor 42Ncm	1

Table 19: Sorted Pill Bin Assembly Parts

Sorted Pill Bin	
Item	Quantity
Stepper Motor 13Ncm	1
Turntable Bearing	1
Neodymium Magnets	6
M5 x 14 mm Hex Bolt	4
M5 x 45 mm Hex Bolt	6
M5 Hex Nut	6

Table 20: Pneumatics

Pneumatics	
Item	Quantity
12V Vacuum Pump	1
SMT Head	1
Vacuum Tube 4mm	1
Vacuum Tube 6.5mm	1
Pneumatic Connector	1
12V Solenoid Valve	1
Relay Module	1

Table 21: Electronics

Electronics	
Item	Quantity
Power Supply	1
Arduino Uno	2
CNC V3 Shield	1
Emergency Stop	1
Fused Power Socket	1
Current Sensor	1
Addressable LEDS	1

For nearly all of the components, outside of those shown above, an FDM 3D printer was used to create the pill bins, sorted pill bins, sorting arm, sorting arm shell, gear mesh plates, linear rail guides, and gears. Of these main assemblies, the sorting arm and sorted pill bin are the most complex to assemble so exploded views have been provided below.

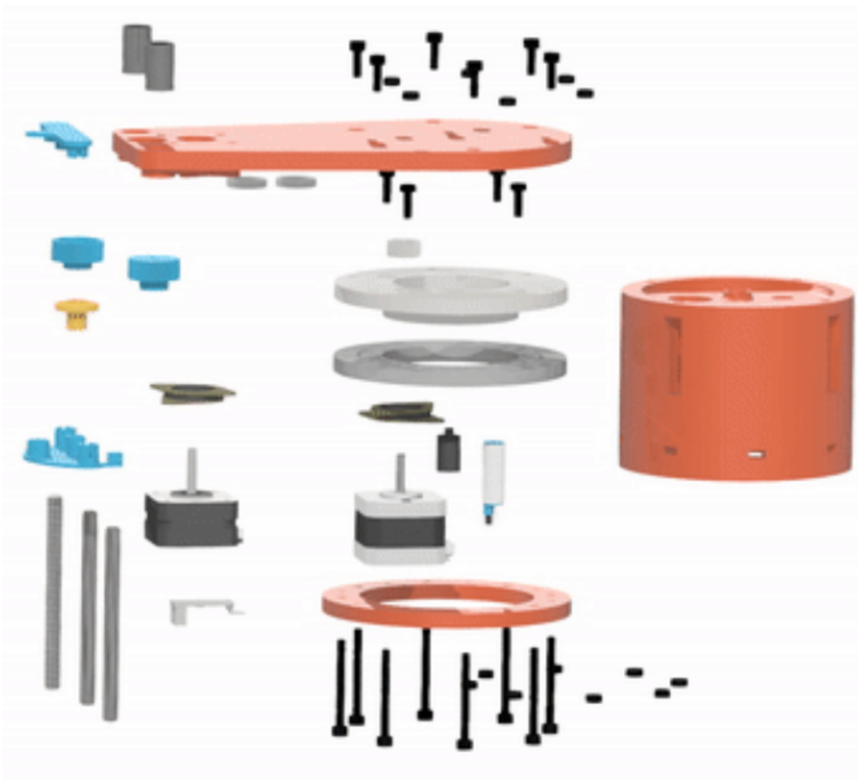


Figure 76: Exploded Sorting Arm

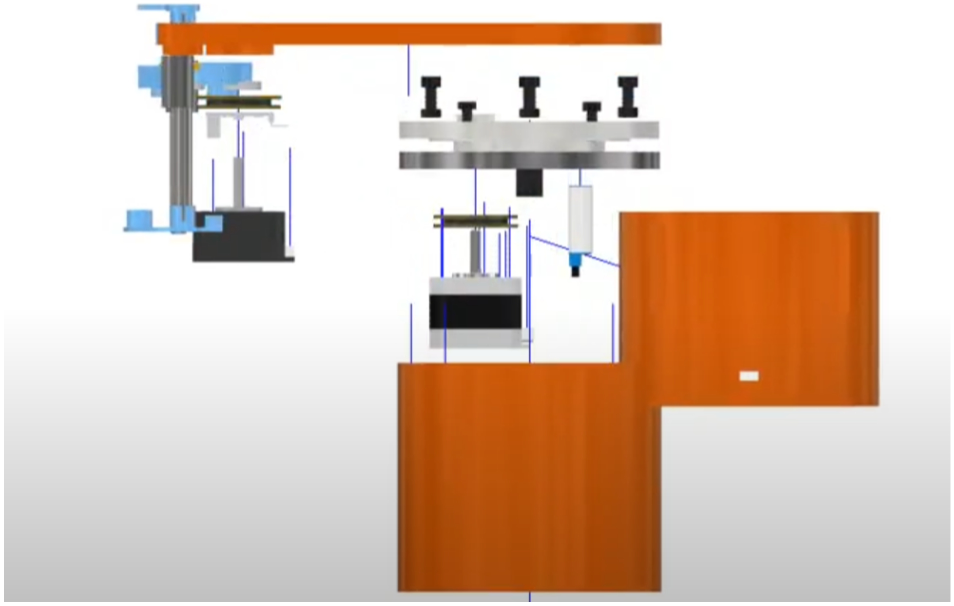


Figure 77: Exploded Sorting Arm

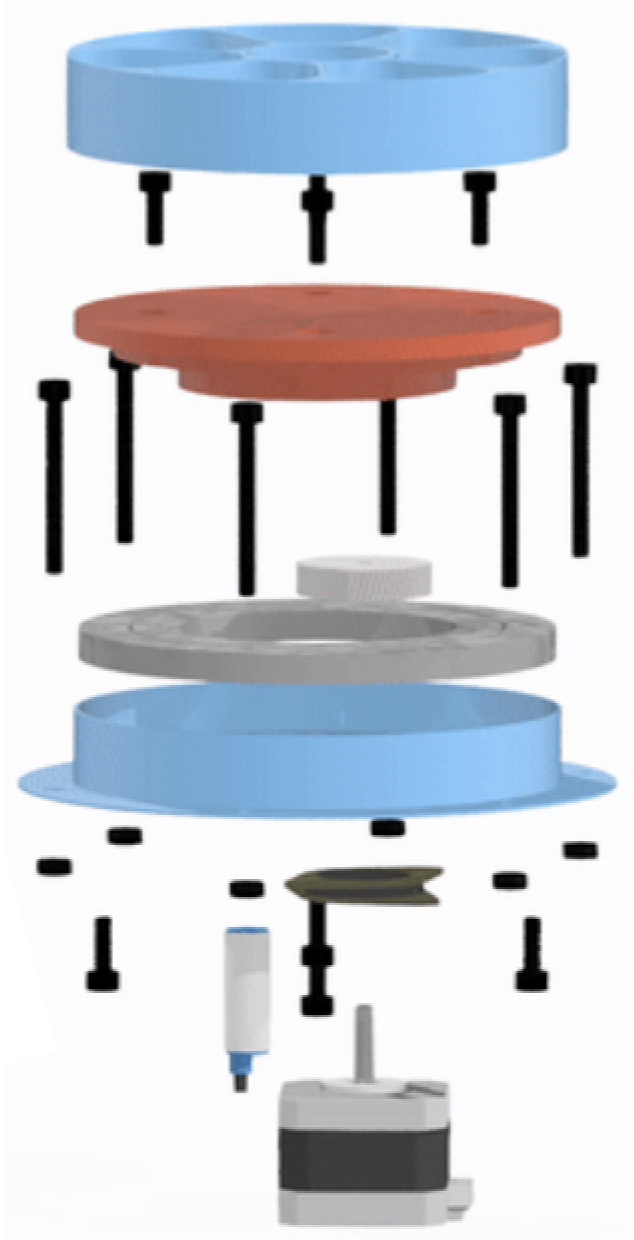


Figure 78: Exploded Sorted Pil Bin

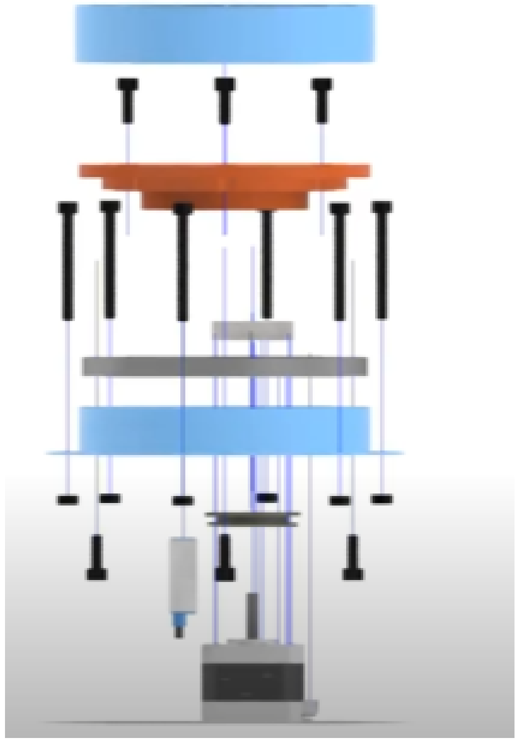


Figure 79: Exploded Sorted Pill Bin

To mount all of these components a circular wooden project panel 1 ½ ft in diameter was used. To ensure that components are mounted accurately and properly the board was cut using a waterjet CNC machine (thanks to Frank Coffey). A layout of the cuts to be made has been provided below.

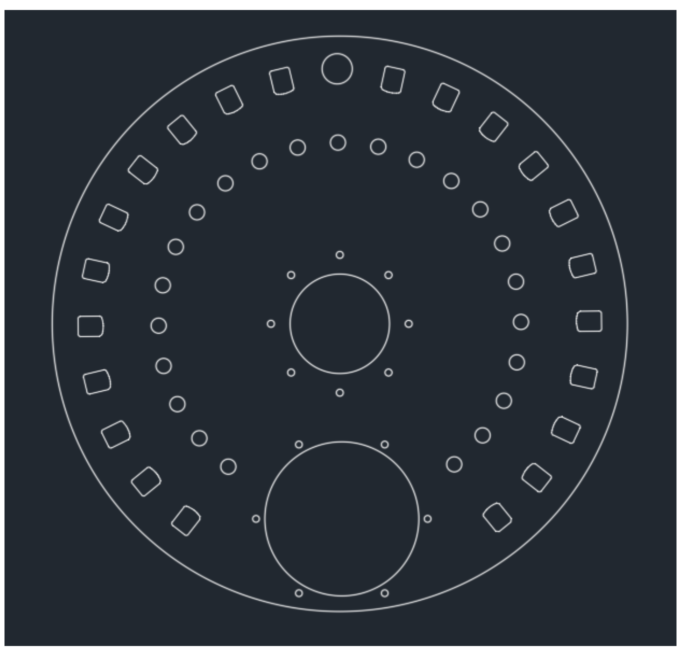


Figure 80: Board Layout

Lastly, pictures of the final assembly are provided below.

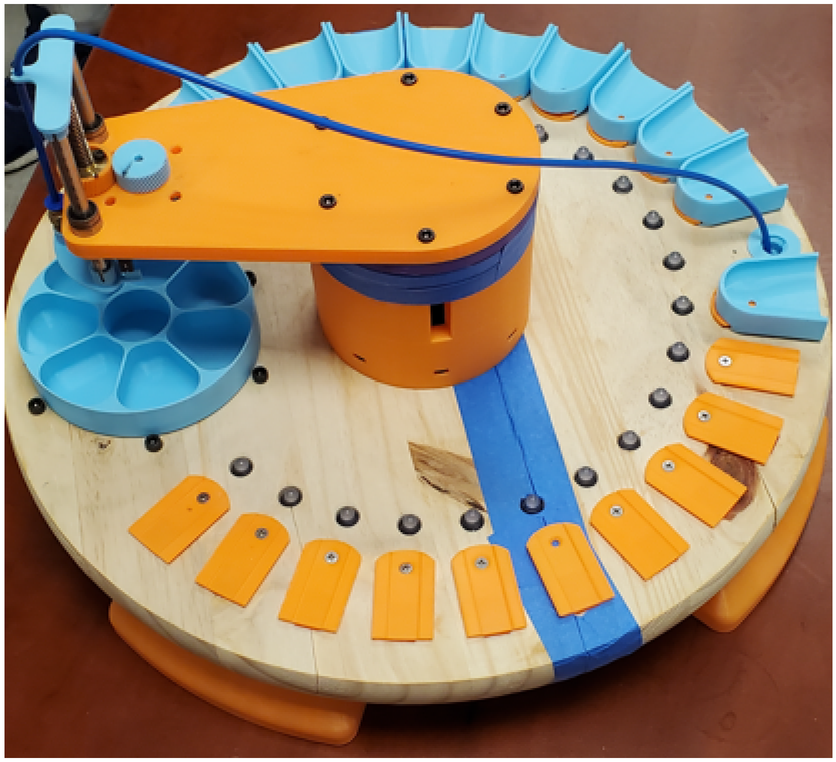


Figure 81: Assembled Top

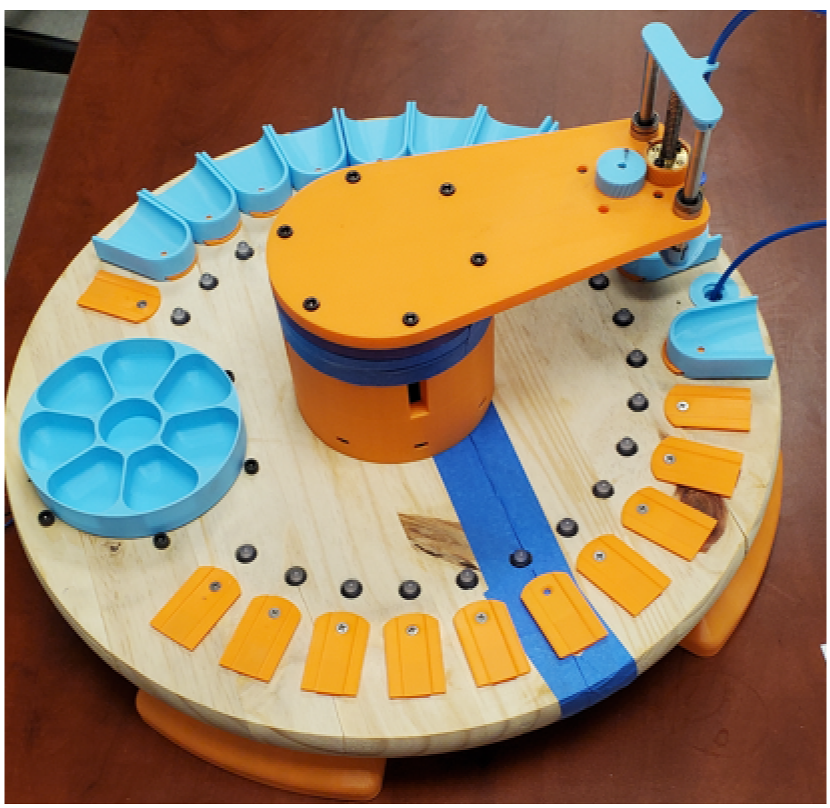


Figure 82: Assembled Top

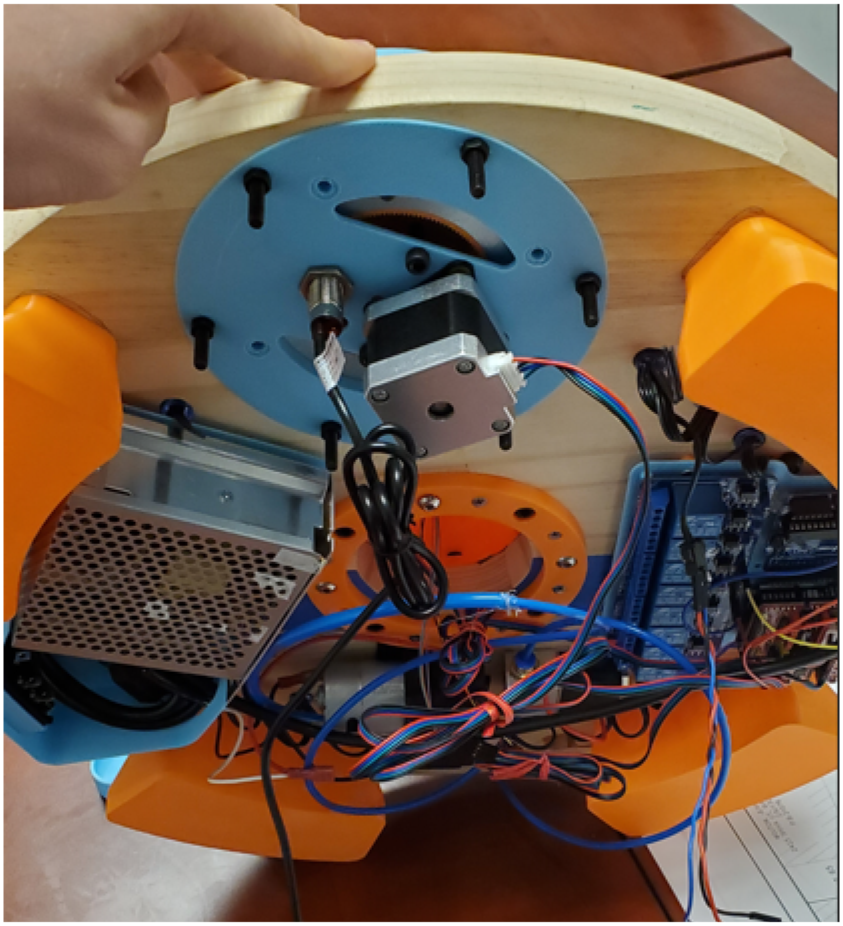


Figure 83: Assembled Bottom

Analysis

Due to this project being concerned with the movement of medication the loads exposed to the machine are incredibly low as the largest pill would weigh only 5 grams. For this reason, the focus of this section will be on the stress analysis of the central head unit when subjected to a transverse load at the end of the pill retrieval arm. Additionally, since the arm is able to rotate freely only a pure transverse load is tested as any load applied otherwise would simply cause rotation. The thought process with this analysis would be to identify if damage would occur to the central unit if a weight were to be placed atop the machine. A visual representation of the model tested and its loading can be seen below.

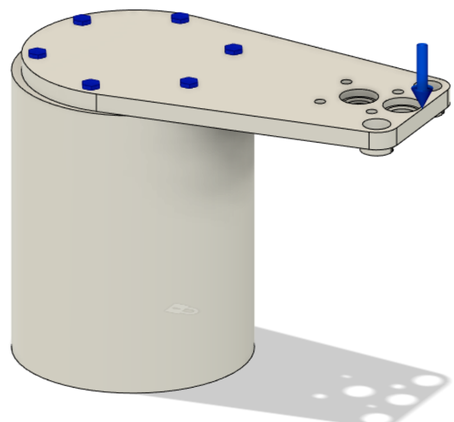


Figure 84: Stress Analysis Setup

In the analysis, loadings were applied at the 10, 15, 20, 30, and 40 kg marks as shown above, and the Von Mises stress, displacement, and minimum safety factor were found. In the following figures below are the results of each of these tests. The most important of all of these figures in identifying the overall strength of the central sorting arm is the figures depicting the safety factor. In general, the safety factor as defined by autodesk should exceed 3 on all parts of the model, but as can be seen in all the tests below this is not achieved. However, as long as the safety factor remains above 1 there will be no permanent deformation to the sorting arm. This is seen as the case up until the 30 kg test at which the model reaches the point that it would see permanent deformation. As can be seen in the 20 kg test the minimum safety factor is 1.014 which is very close to that 1.0 limit. Additionally, this means that the central sorting arm will be able to withstand a maximum of a 20 kg load, leading to a max displacement of 5.983 mm and a max Von Mises stress of 19.72 MPa. Extending beyond this 20 kg limit in the analysis allows for the point at which the plastic will deform or break to be easily seen as shown in the 40 kg tests. In the 40 kg tests, there is clearly a highlighted region in the Von Mises stress and stress factor figures where the arm will fail indicated by the yellow/red regions.

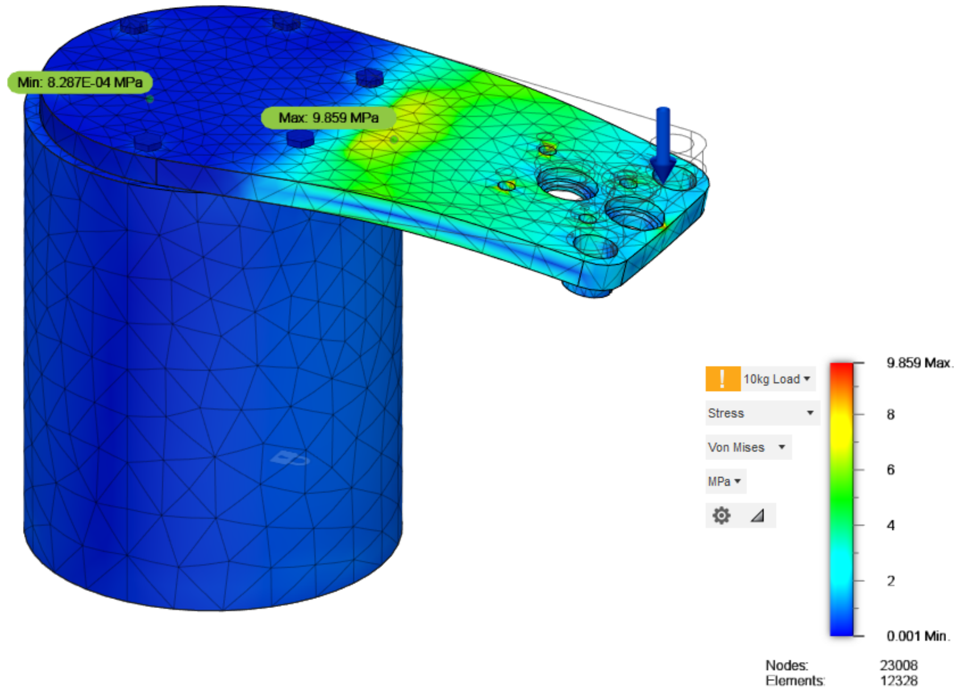


Figure 85: 10 kg Von Mises Stress

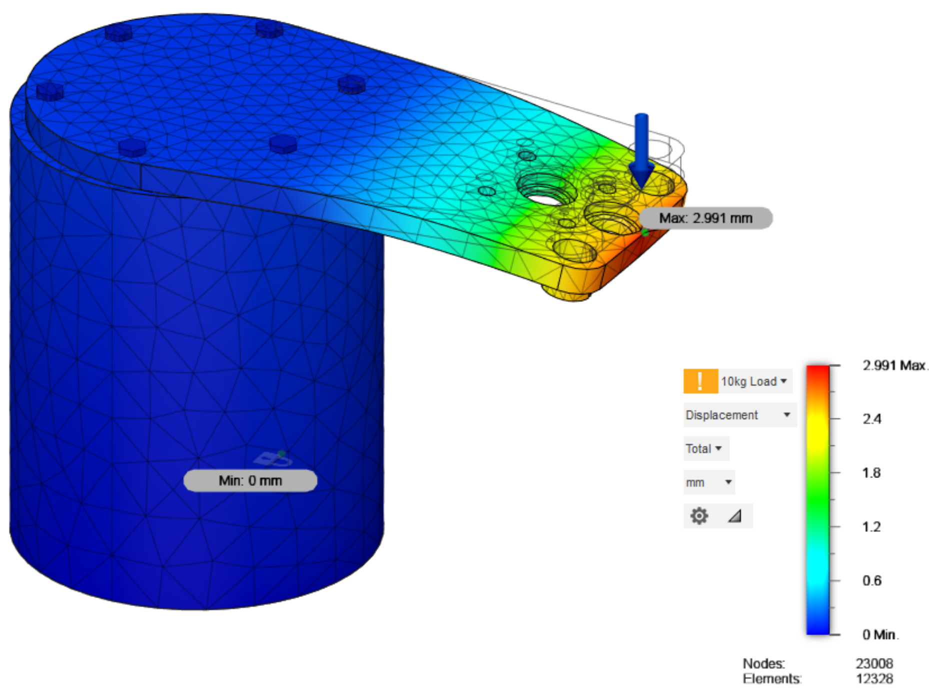


Figure 86: 10 kg Displacement

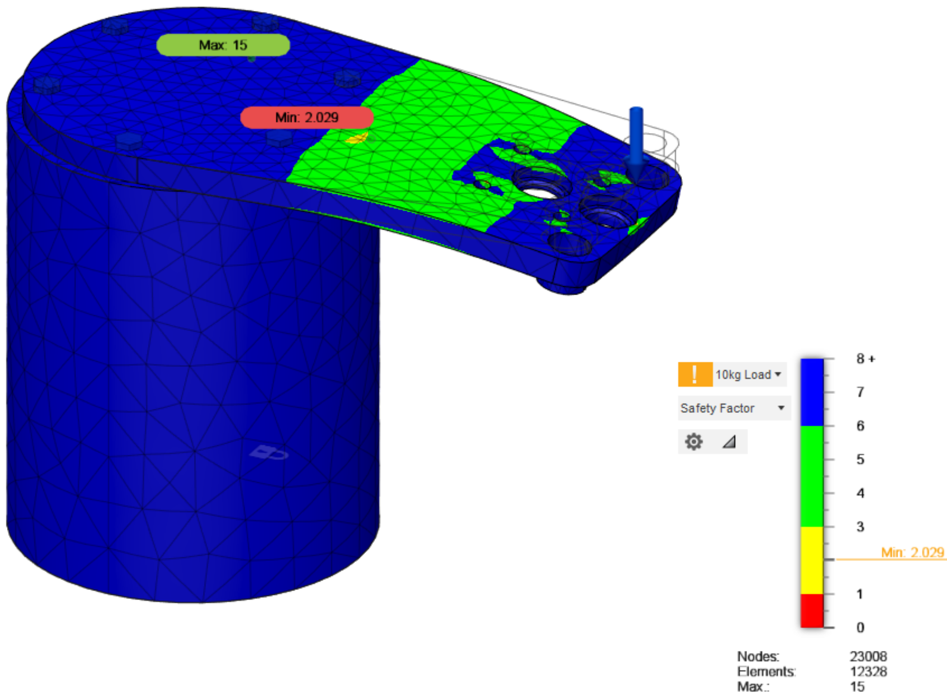


Figure 87: 10 kg Safety Factor

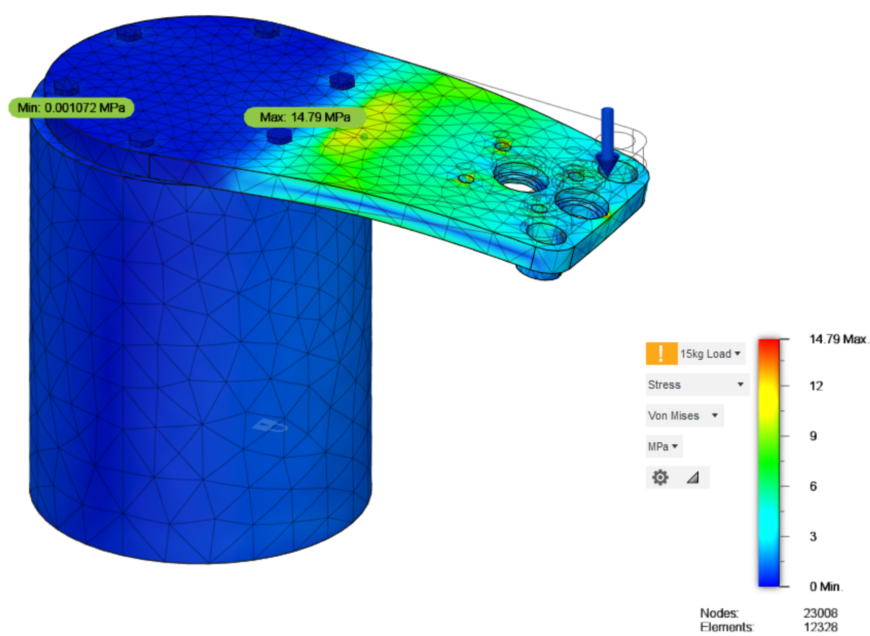


Figure 88: 15 kg Von Mises Stress

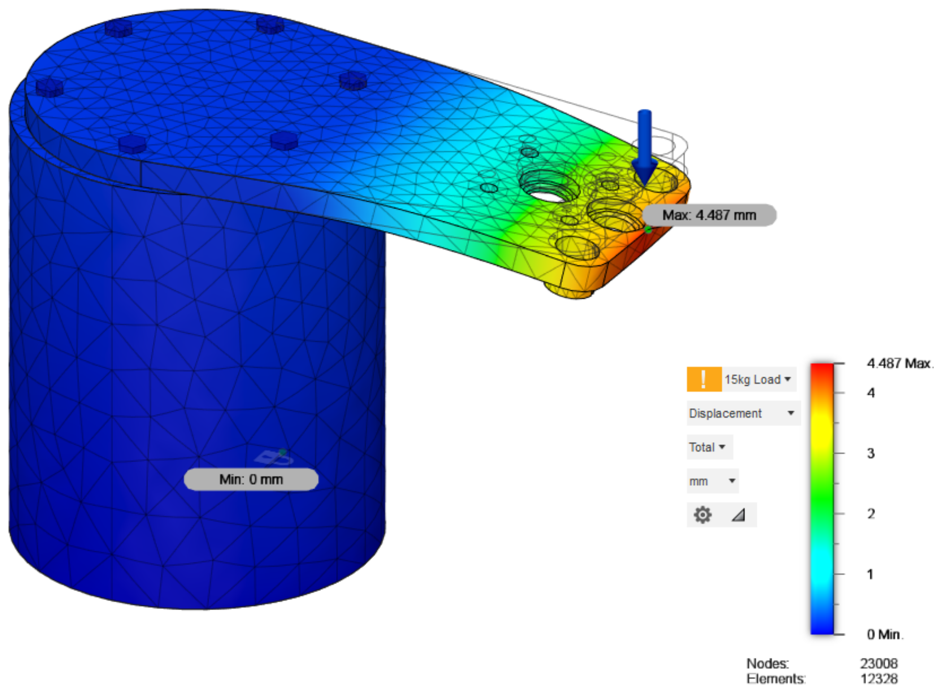


Figure 89: 15 kg Displacement

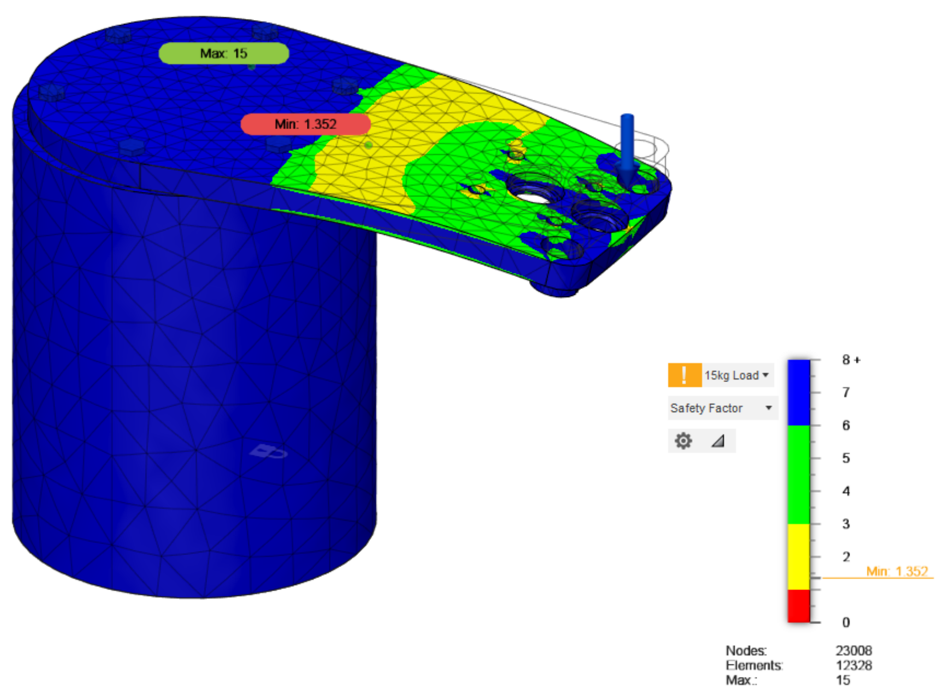


Figure 90: 15 kg Safety Factor

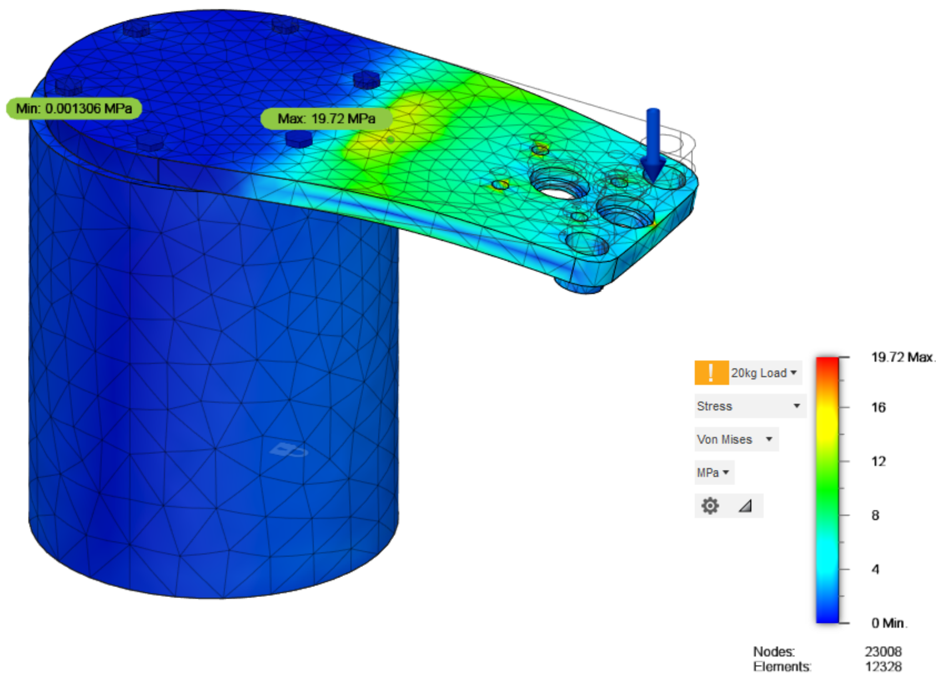


Figure 91: 20 kg Von Mises Stress

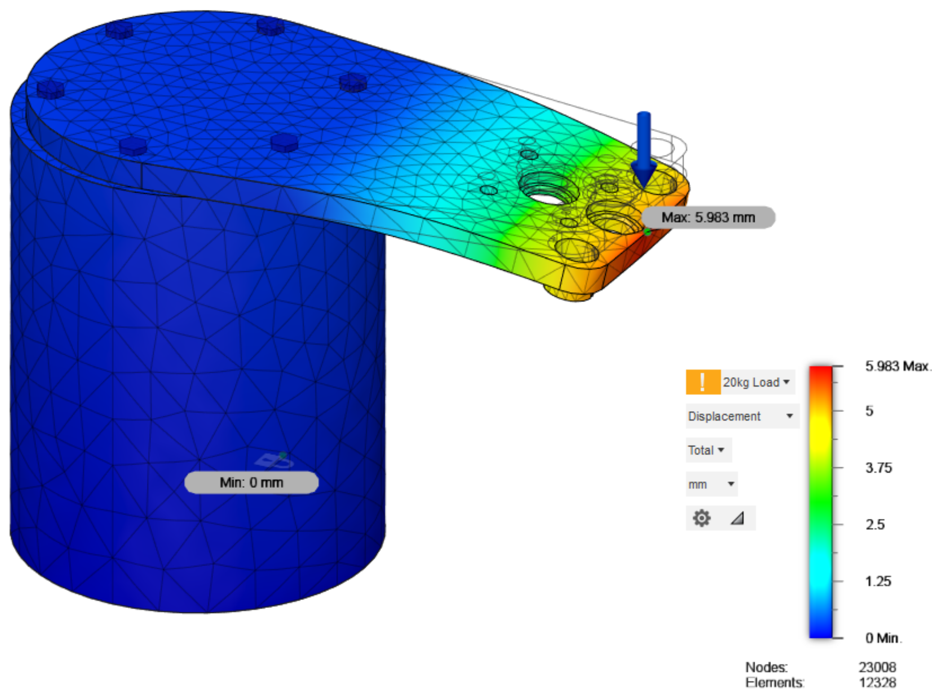


Figure 92: 20 kg Displacement

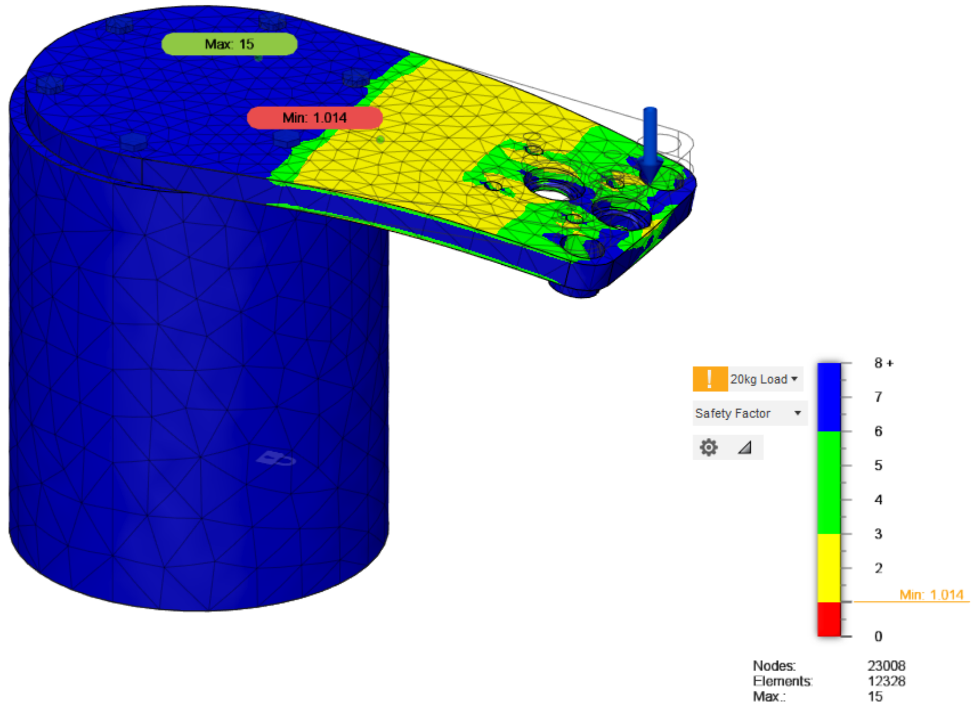


Figure 93: 20 kg Safety Factor

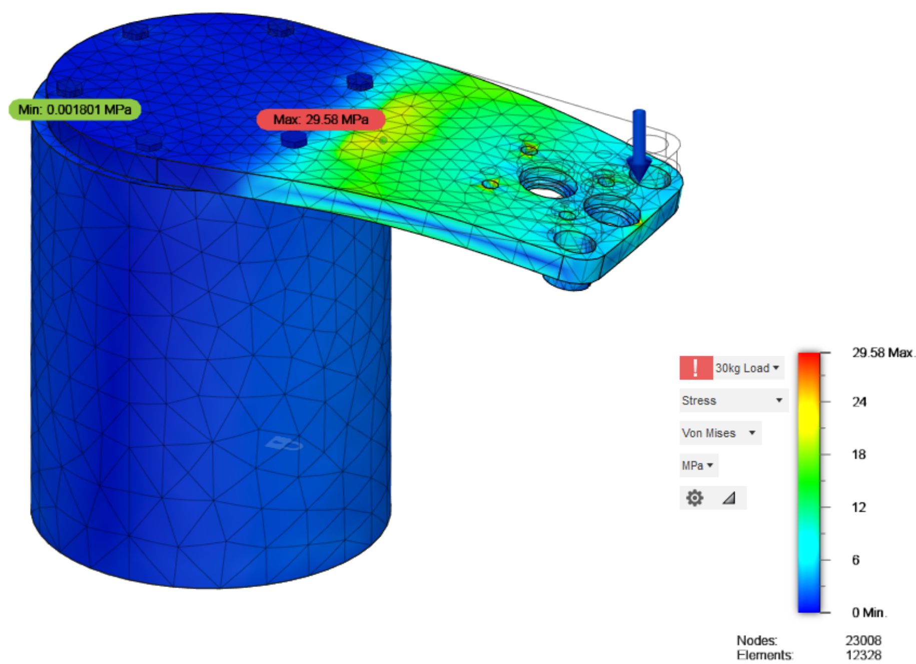


Figure 94: 30 kg Von Mises Stress

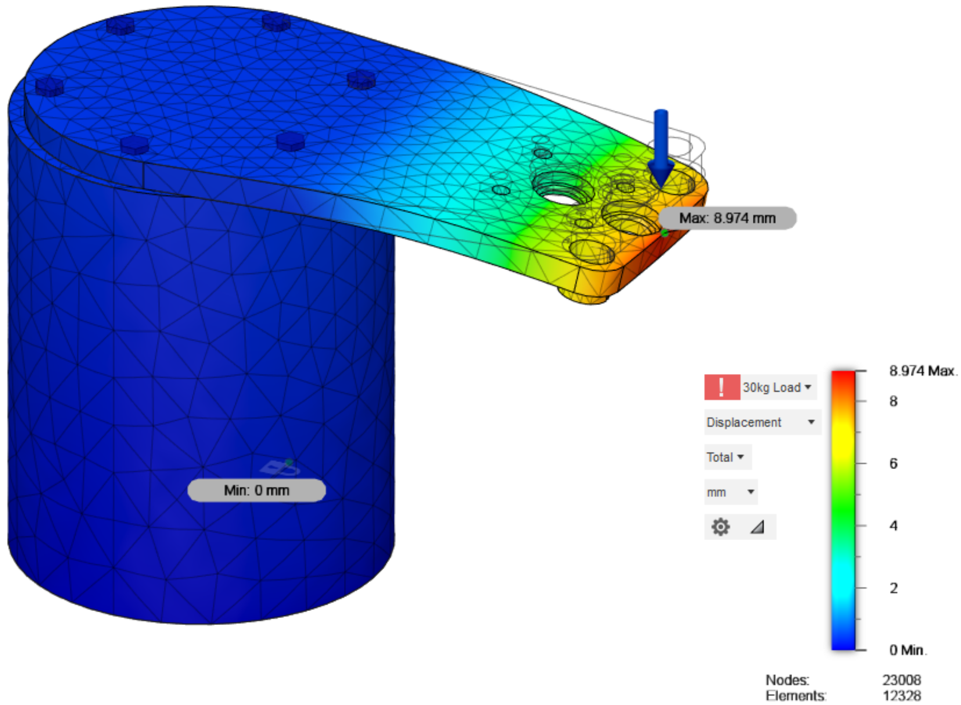


Figure 95: 30 kg Displacement

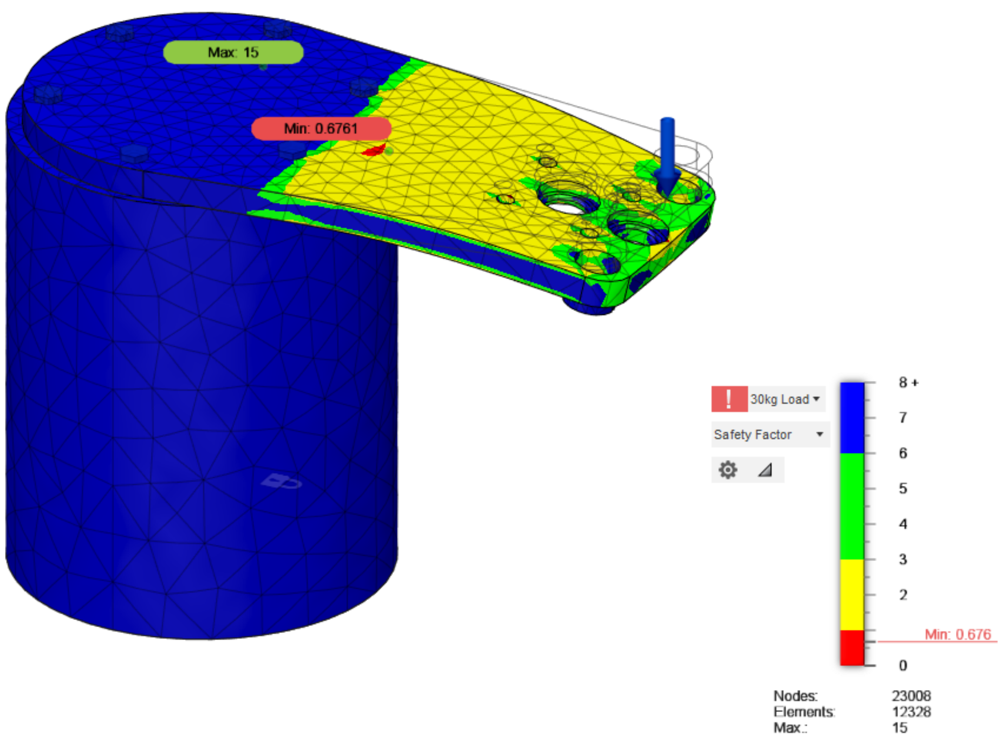


Figure 96: 30 kg Safety Factor

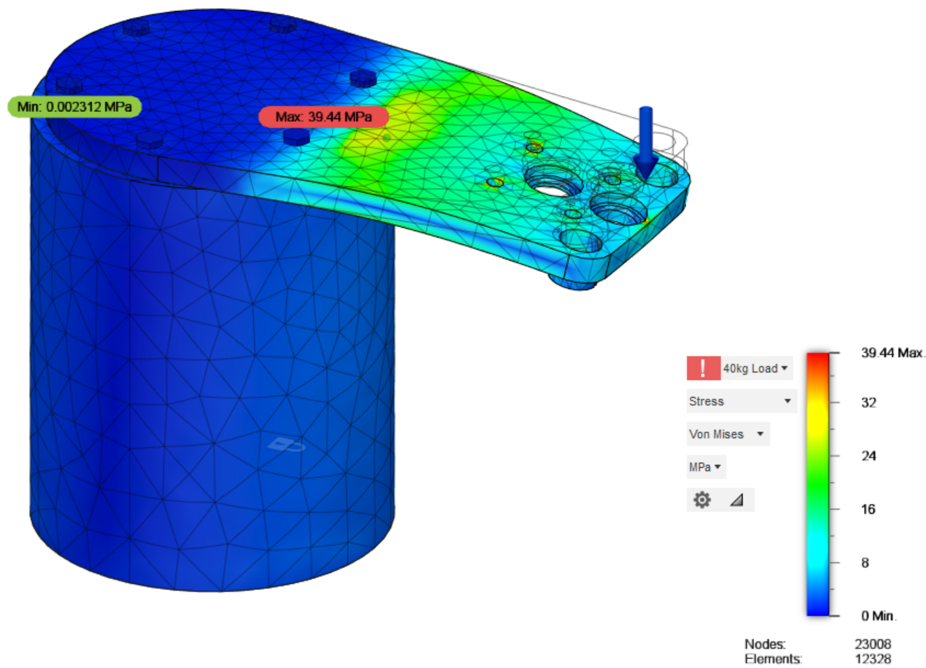


Figure 97: 40 kg Von Mises Stress

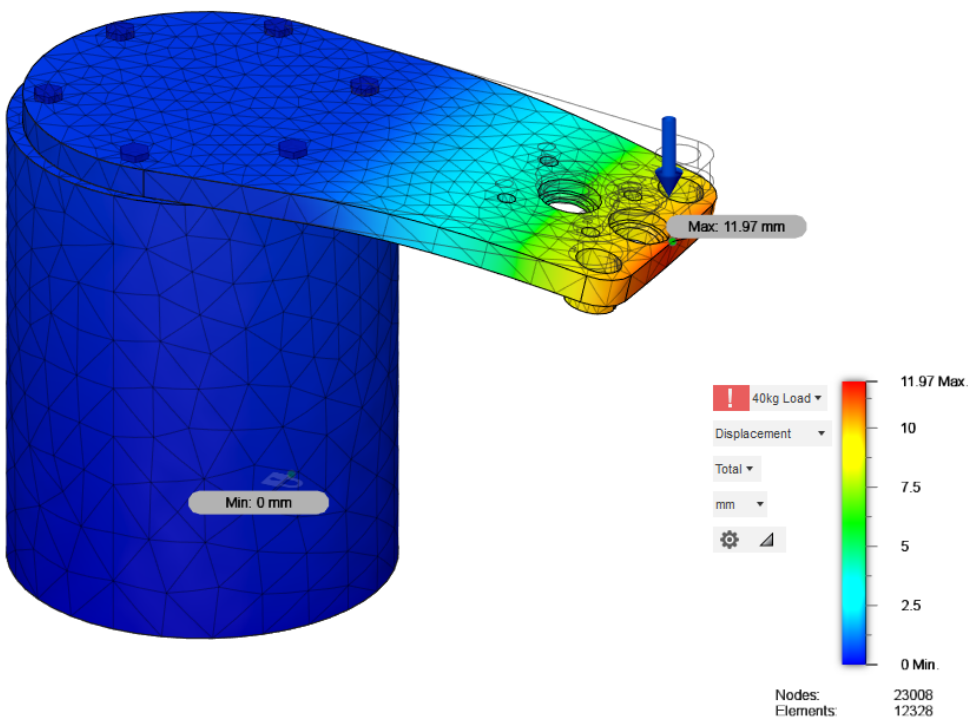


Figure 98: 40 kg Displacement

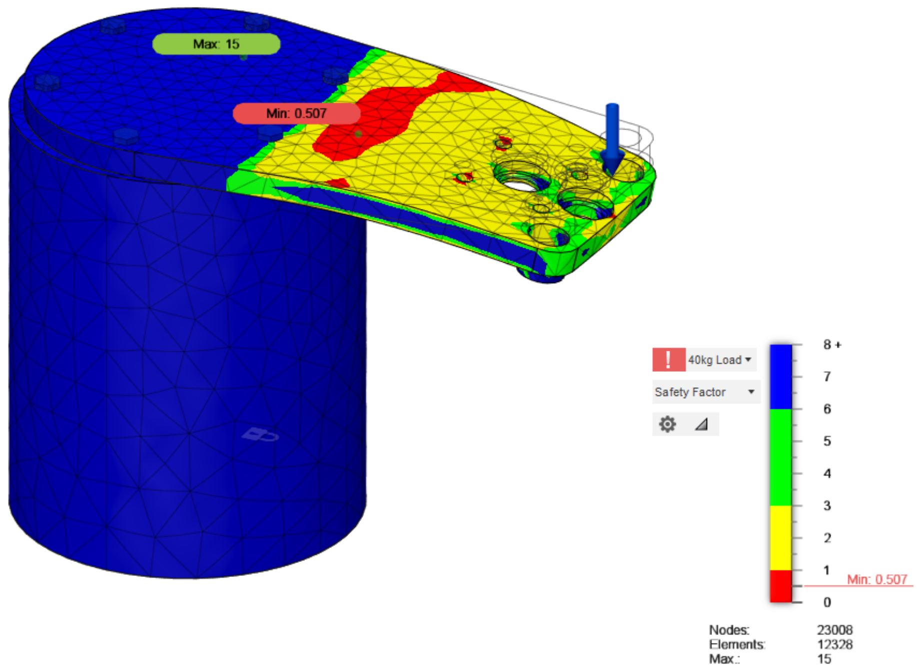


Figure 99: 40 kg Safety Factor

Software

For this project the vast majority of the problem of sorting pills correctly is solved at the software level. In sorting medications correctly, all that needs to be done is for each patient to have their medications and their corresponding dosage/frequency entered into a database correctly by the prescribing physician. By making the physician responsible for entering the medications information into a database it will prevent strain on a technician operating the proposed sorting device as well as ensure accurate information. With the patient's prescribed pharmaceutical drug routine provided, all the technician will need to do is to select the patient's order, load the medication, and let the machine sort the medications.

In controlling the sorting device, a single Arduino UNO loaded with the open source CNC software GRBL is sufficient for making simple movements based on G-Code read from a file. The issue with sorting the medication based on commands read from a file is that GRBL is not smart enough to perform backtracking or know when it needs to stop (unless it hits an end stop) as it simply reads G-Code commands and executes them. This necessitates an additional Arduino UNO that is capable of controlling the RFID reader, LEDs, and current sensor. Additionally, an Arduino UNO is only a microcontroller so it is not capable of creating the graphical user interface required for this project. This means that in order to control both the Arduino UNOs and the graphical user interface some programming language is needed.
In selecting the programming language to create the graphical user interface and issuing commands to the two Arduino boards Python was selected. To simplify the process of creating

the graphical user interface the library PyQt is used which is a project based in C++ that was ported to Python. By using this library the process of creating the GUI is greatly simplified while also improving process speed and reducing bugs. Finally, to interface with the two Arduino boards the Python library PySerial is used which allows for direct control of the boards over a serial connection. Now, in addition to Python the project uses some C++ which is the language that Arduino uses. On the Arduino Uno responsible for the control of the LEDs, RFID reader, and current sensor, native Arduino libraries are used in addition to Adadfruit's neopixel library for LED control.

Located in the figure below is a UML diagram of the program and its multiple interfaces. At the core of the program is the main Pill Sorter interface which takes in a medication object containing all of a patient's medical information. The Pill Sorter interface interfaces with both Arduino UNO boards and allows for the creation of the Direct Control interface, Config interface, and the Sorting Pill Dialog interface. For the Direct Control and Sorting Pill Dialog interfaces there are worker threads that allow for their respective interface to continually update during operation.

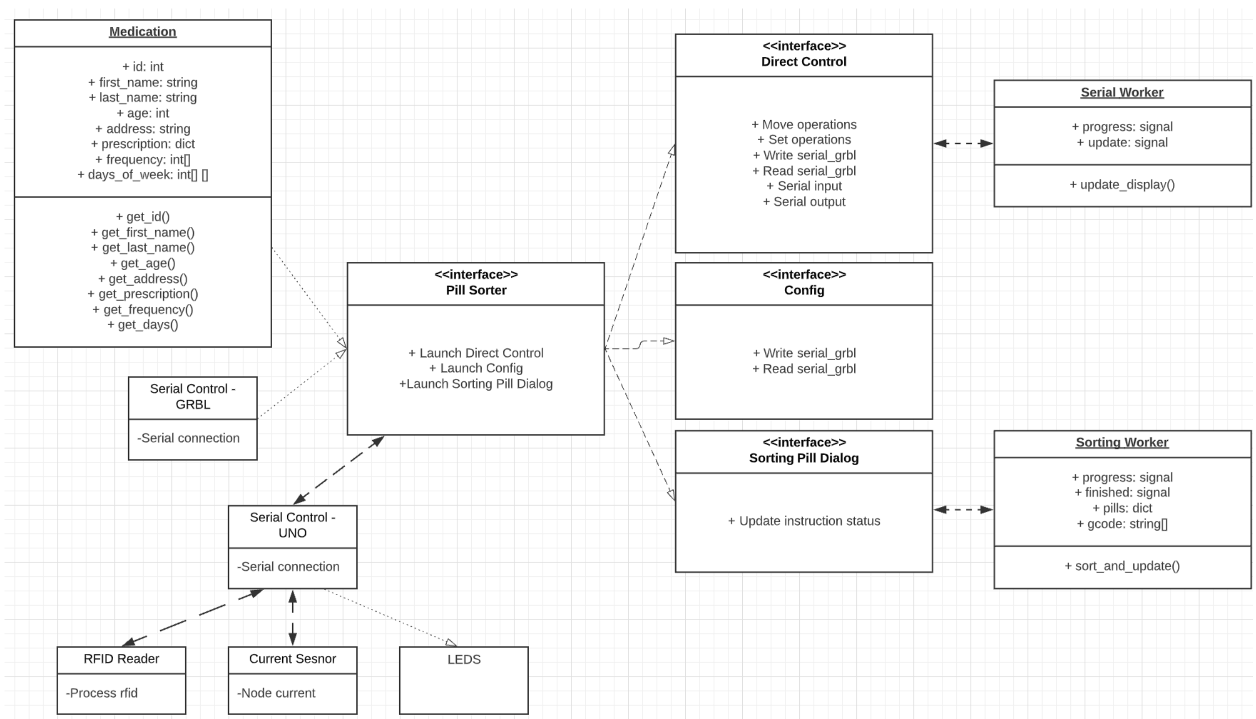


Figure 100: UML Diagram

Now to control the Arduino board running GRBL, G-Code commands need to be created to move the servo motors of the machine. In its current state the software generates all the G-Code commands at runtime when the operator selects a patient's medication to fill. While sorting, the software listens to signals from the secondary Arduino for an alarm state indicating that a pill has been picked up or dropped. In the event of the picker head descending and not obtaining a pill the sorting process backs up several steps and attempts to grab a pill again in a location a few millimeters away from the initial attempt. Finally, in order to control the suction of the picker head the Spindle pin on the GRBL board has been repurposed to control the state of the solenoid valve. For reference of basic G-Code statements see the figure below or visit https://gcodetutor.com/cnc-machine-training/cnc-g-codes.html.

```
G21
M1403
M1405 X180.02 Y-145.06
M1406 X263.16 Y-161.58
M1407 S20.0
M1410 1.4
G0 X206.29 Y-153.32
M3
M8
G4 S20.
```

Figure 101: G-Code Example

In the following figures are pictures of the graphical user interface and its utilities accompanied by some explanations. In the first of the three figures below is the initial screen presented to the user when the program is launched. This initial screen allows for the user to launch the configuration and direct controller interfaces as well as select a prescription from the database.

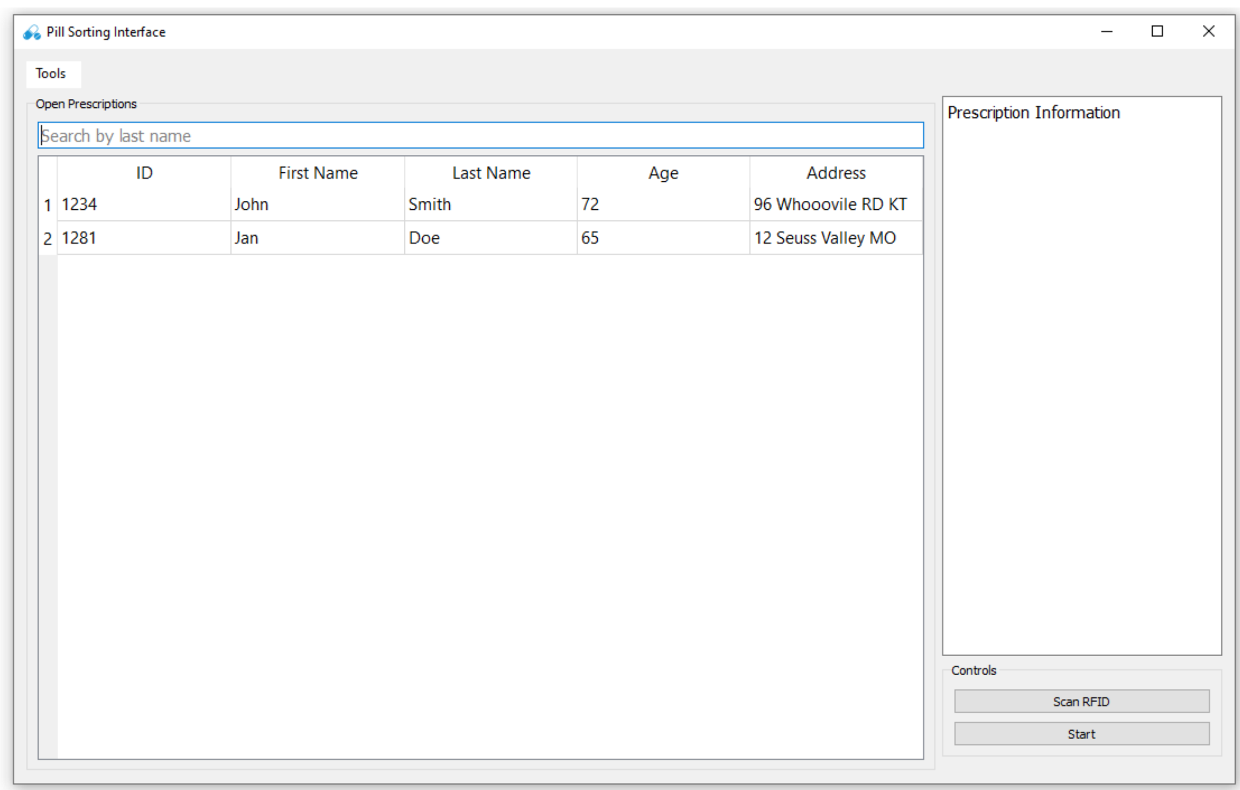


Figure 102: Initial Selection Window

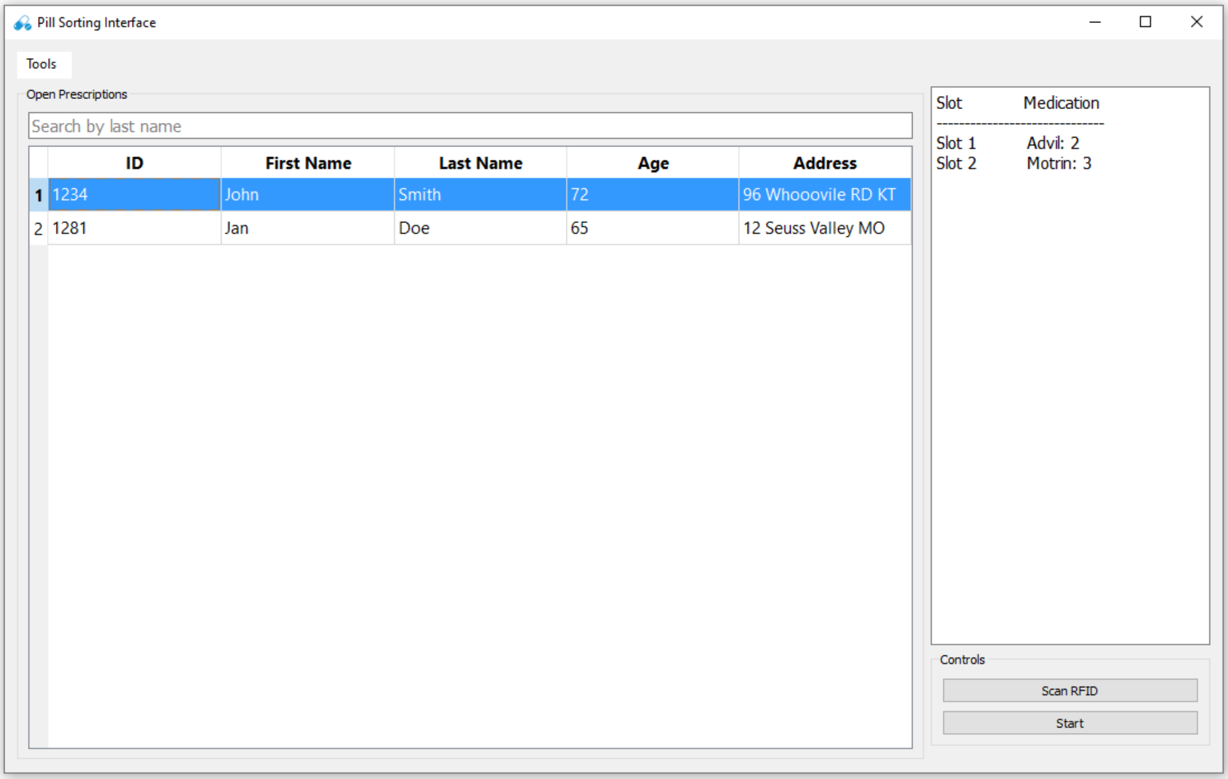


Figure 103: Selection Window with Selection

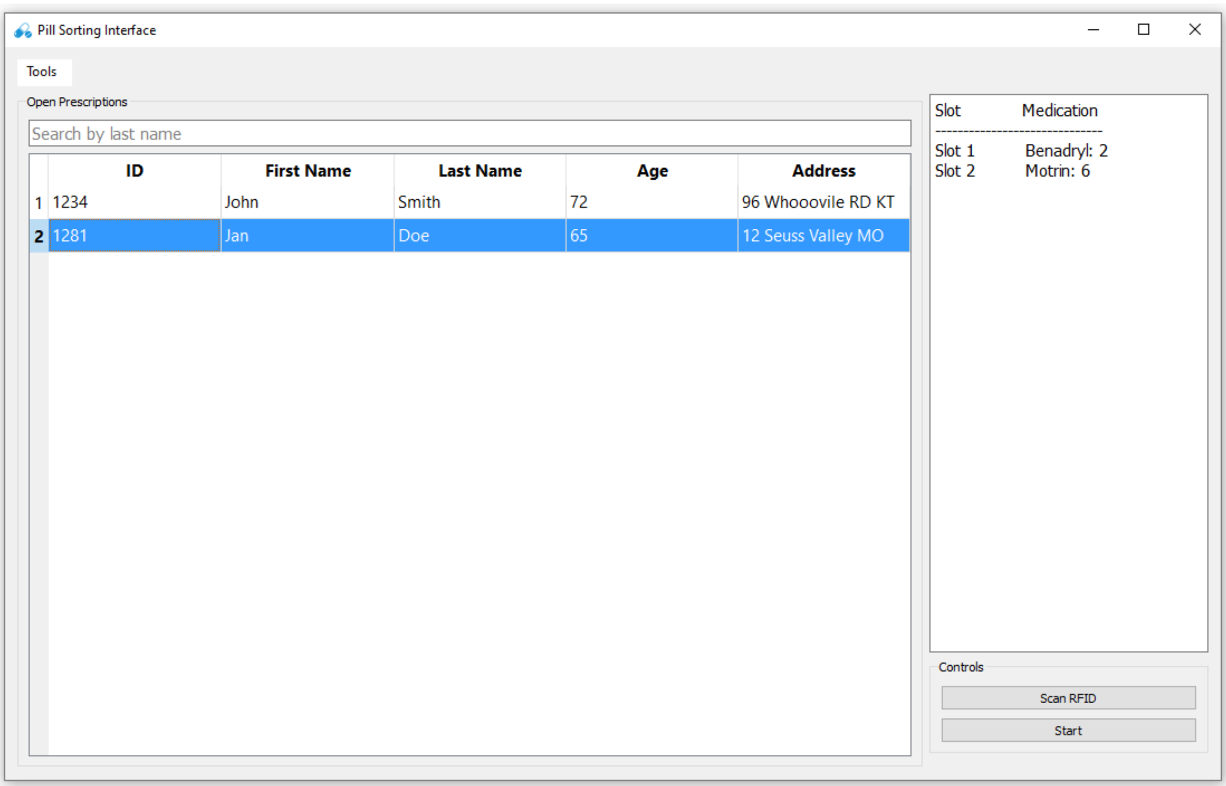


Figure 104: Selection Window with Selection

The following figure shows the configuration interface which allows for the user to modify the operation characteristics of the machine.

Figure 105: Configuration Tool

The following two figures show the direct controller interface which allows for the setting of positions, relative movement control with the various x +10 y +1 buttons, absolute movement with the spinner boxes, and finally direct commands via G-Code using the serial input field. As can be seen in the second of the two figures, the serial output field is a direct output of what the Arduino running GRBL is communicating.

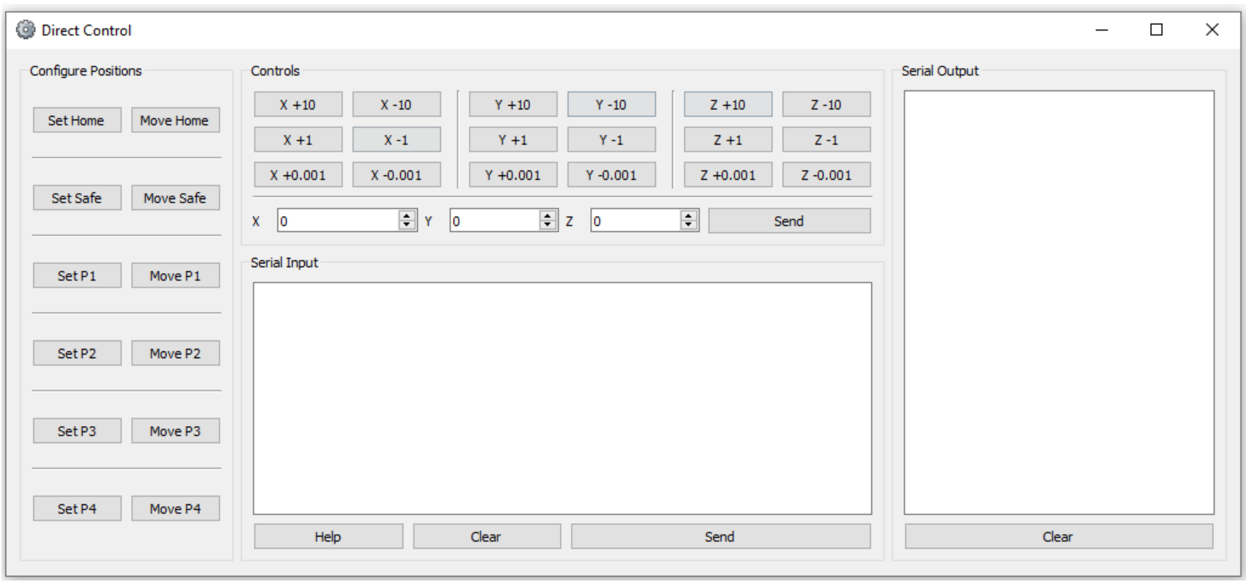


Figure 106: Direct Control Tool

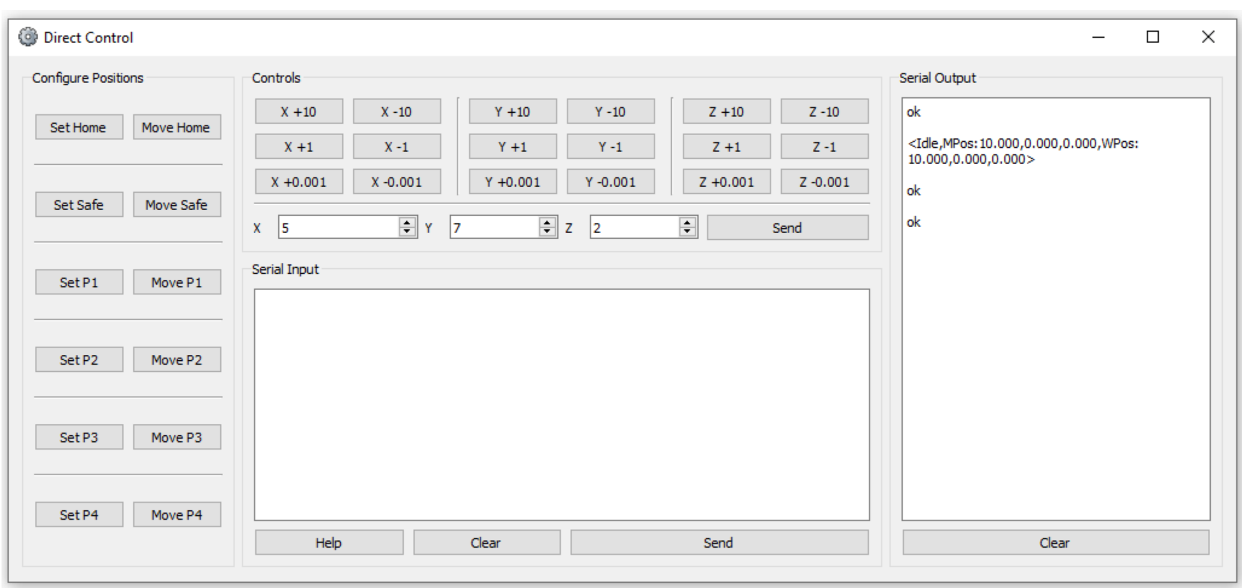


Figure 107: Direct Control Tool with Commands Entered

In the final four figures are the various stages of the sorting process. First off, a message box pops up reminding the user where medication is to be loaded along with the option to do the process in steps to ensure proper sorting if necessary. Once the user presses ok a countdown message is presented indicating the sorting is about to begin. Next, a new message is displayed along with a progress bar indicating the point that the sorting process is at. Lastly, a message box is presented to the user indicating the time taken to complete the sorting process.

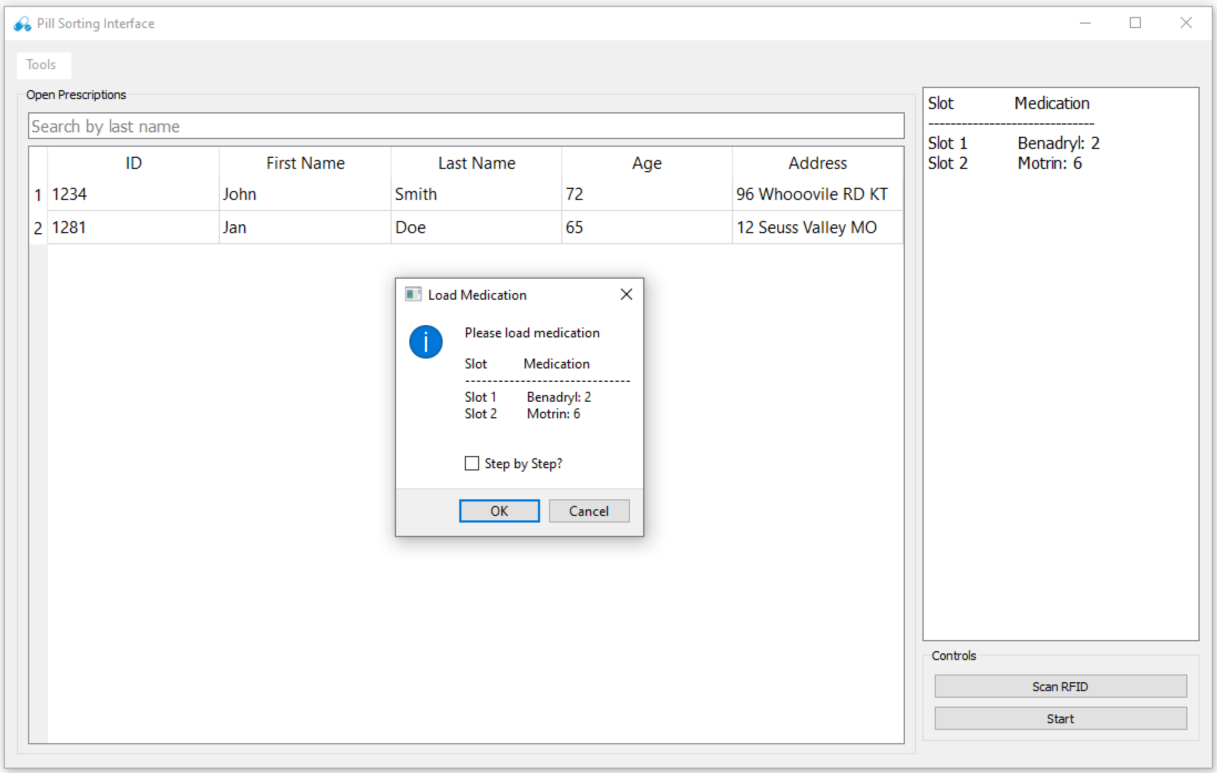


Figure 108: First Sorting Message

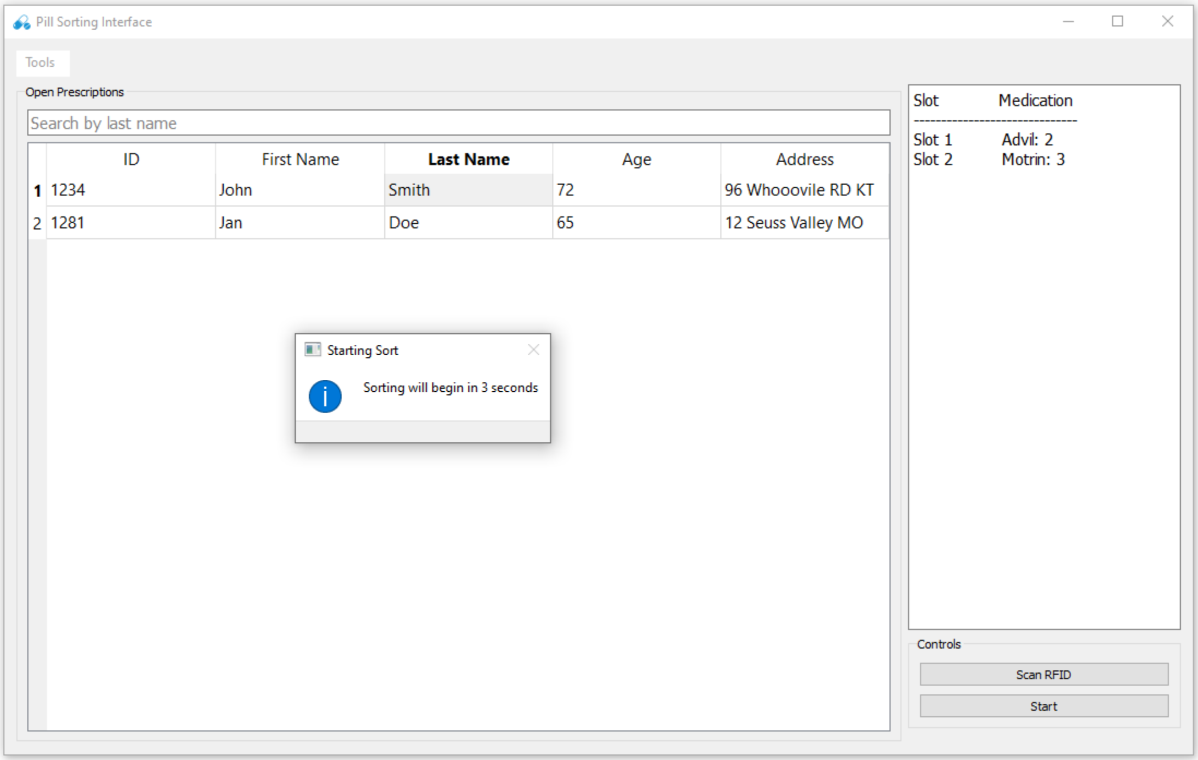


Figure 109: Countdown for Sorting

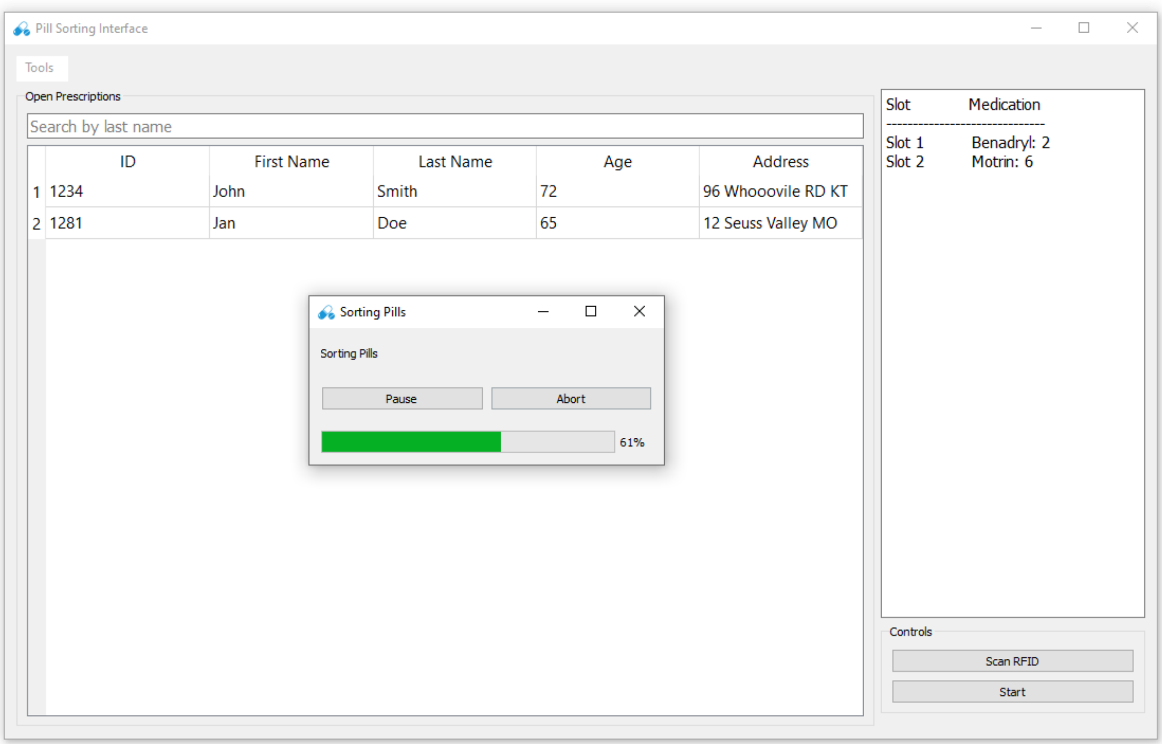


Figure 110: Sorting Progress

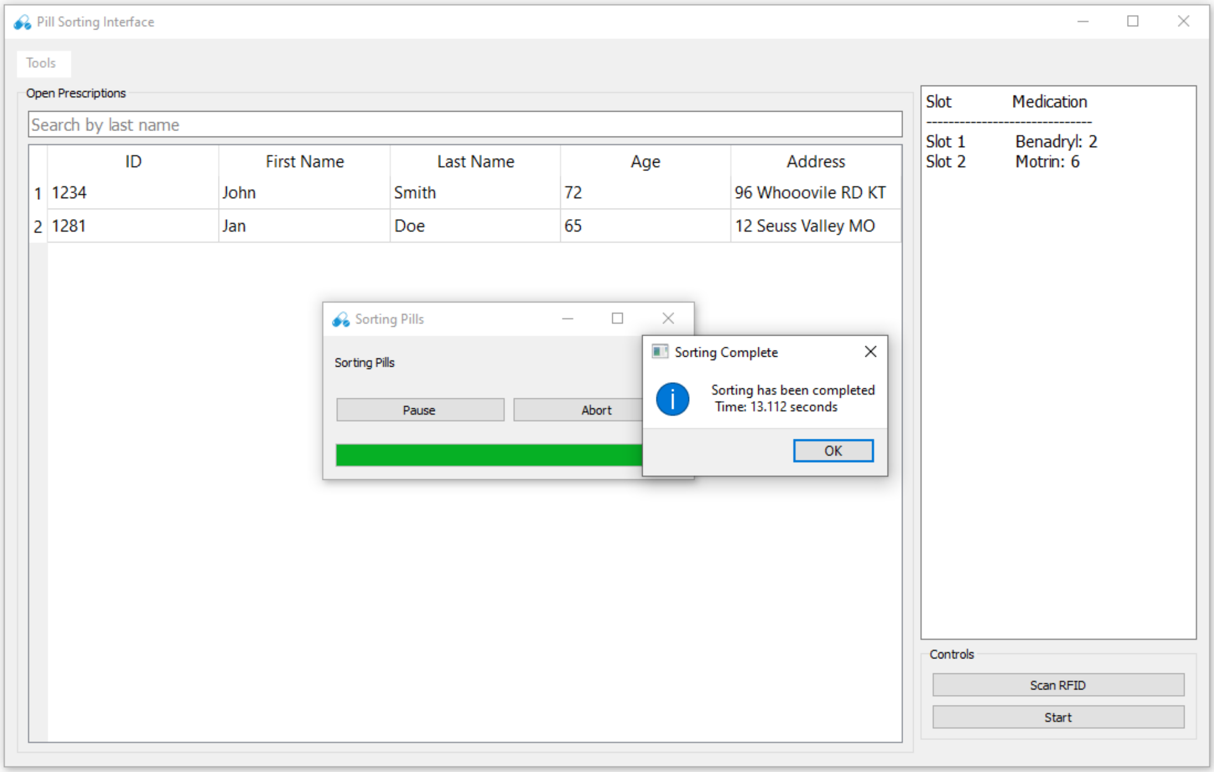


Figure 111: Sorting Complete

Test Plan & Verification

For the testing and verification of the assembled prototype a multi-phase testing model will be used. At the first stage a single pill storage bin will be sorted into the sorted pill bin using multiple different sizes of 'pills'. Note that for testing, only over the counter medications will be tested but some various hard candies may be used as well. At the end of each trial of stage one the run time and accuracy will be recorded by verifying that the correct number of pills have been sorted and they are in the correct sorted locations. For the second stage of testing two different medications will be sorted into the sorted bins and the same measurements taken as in stage one. In the third stage, the stage that will be used in demonstration, six pill storage bins containing the same medication will be sorted into the sorted bin and the same measurements taken once again. Finally, stage four will be a repeat of stage three but with different medications in the six pill storage bins. Note that the device has more than six slots available which based on testing may be tested as well in a final testing phase. Located in the table below is an outline of how testing will be conducted.

Table 22: Proposed Test Plan

Phase	Bins	Types of Medication	Repetitions	Time	Accuracy
1	1	1	5	?	?
2	2	2	5	?	?
3	6	1	5	?	?
4	6	6	5	?	?

As can be seen in the table above only 5 repetitions are currently being proposed for the testing process, but this number is likely to increase depending on the amount of time available. Ideally, to be statistically sound each test should consist of ~30 repetitions to achieve an accurate estimation of the machine's true accuracy. Once testing has been completed, the obtained values by measuring and calculating the time, accuracy, and precision shall be compared to those outlined earlier in the requirements section. One final piece of analysis that could be conducted if necessary, would be to perform a two-level factorial design to identify what variables such as pill shape and size play on the machine's performance.

Located in the following four tables are the results of these tests. As can be seen in testing phases 2 and 4, accuracy is much lower than in phases 1 and 3. This loss in accuracy is primarily attributed to the large variation in pill properties which presents itself more as the types of medication tested increases. In particular the machine struggles greatly with capsule style pills as when the pill is touched by the suction tip a solid seal is not made, causing the grab location to shift to the end of the pill rather than the middle. Additionally, some pills will often get pushed out of the way of the suction tip rather than the tip making contact and capturing the pill. Lastly, as the sorting arm gets to further out pill locations the arm ends up roughly 1mm off the desired location due to warping that occurred when using the waterjet to cnc the board.

Table 23: Testing Phase 1

Phase 1					
Trial	Bins	Types	Pill Count	Time	Accuracy
1	1	1	14	148.719	85.71%
2	1	1	14	148.509	92.86%
3	1	1	14	148.725	71.43%
4	1	1	14	148.651	78.57%
5	1	1	14	148.768	85.71%

Table 24: Testing Phase 2

Phase 2					
Trial	Bins	Types	Pill Count	Time	Accuracy
1	2	2	10	106.963	60.00%
2	2	2	10	107.142	70.00%
3	2	2	10	106.876	60.00%
4	2	2	10	106.911	40.00%
5	2	2	10	106.974	60.00%

Table 25: Testing Phase 4

Phase 3					
Trial	Bins	Types	Pill Count	Time	Accuracy
1	6	1	24	253.104	83.33%
2	6	1	24	253.504	70.83%
3	6	1	24	253.301	79.17%
4	6	1	24	253.336	75.00%
5	6	1	24	253.612	83.33%

Table 26: Testing Phase 4

Phase 4					
Trial	Bins	Types	Pill Count	Time	Accuracy
1	6	6	24	253.308	50.00%
2	6	6	24	253.278	33.33%
3	6	6	24	253.512	41.67%
4	6	6	24	253.289	37.50%
5	6	6	24	253.502	54.17%

Next, the results from the four test phases have been averaged and can be seen in the table below. As can be seen, the accuracy of the machine is respectable when only 1 type of pill is used, but is still far off the 99.99% that was the goal. When the types of pills are greater than 1 the machine does quite poorly due to the many reasons stated above. Additionally, the time taken to sort each pill is roughly 10.5 seconds which is much slower than the 0.5 seconds needed to sort 1800 pills/hr. Now, it should be stated that the prototype has to operate as ⅖ of max speed and ½ of max acceleration due to slipping occurring from the use of 3D printed gears. Unfortunately, this would still only get the machine to ~4 seconds per pill giving 900 pills/hr which would be half of what the goal was. Overall, the machine does well with a single type of medication but the accuracy is still too low, and sorting occurs much too slow to be the best choice for a pharmaceutical application. For this reason this project would need further development and iteration before being viable as it could jeopardize a consumer's health and pose a liability.

Table 27: Average Test Results

Phase	Bins	Types	Pill Count	Average Time	Average Accuracy
1	1	1	14	148.674	82.86%
2	2	2	10	106.973	58.00%
3	6	1	24	253.371	78.33%
4	6	6	24	253.378	43.33%

Prototype Cost

Located in the table below is the entire parts list for this project excluding the cost of two arduino uno boards which can be found for ~$10. One point to be made about this project is that this can be completed much cheaper if not greatly restricted by time as it would allow for components to be purchased at OEM prices rather than from consumer ecommerce sites such as amazon.

Table 28: Prototype Costs

Name	Status	Catagories	Quantity	Price	Shipping	Est. Delivery	Arrived	Link
ORDER 1 - Submitted								
Power Supply	Completed	China/12v/120w	1	$12.46	Free	3/12	Yes	https://www.aliexpress.com
12 Vacuum Pump	Completed	NULL	1	$16.46	Free	03/03	Yes	ht
CNC V3 Shield	Completed	NULL	1	$10.84	Free	3/11	Yes	ht https://www.alie
SMT Head	Completed	BT1040ZSGA	1	$133.00	Free	03/06	Yes	https://www.aliexpress.com
Vacuum Tube 4mm	Completed	China/Blue/6x4mm-10m	1	$6.43	Free	03/12	Yes	https://www.aliexpress.com
Vacuum Tube 6.5mm	Completed	Blue/10x6.5mm-3m	1	$5.96	$0.91	03/12	Yes	https://www.aliexpress.com
Emergency Stop	Completed	NULL	1	$1.12	$1.05	03/28	Yes	https://www.aliexpress.com
Fused Power Socket	Completed	Red	1	$0.92	Included	3/28	Yes	https://www.aliexpress.com
Pnumatic Connector	Completed	PY/6mm	1	$0.51	Free	3/28	Yes	https://www.aliexpress.com
Juki Nozzle Holder	Completed	NULL	1	$10.50	Free	03/06	Yes	https://www.aliexpress.com
ORDER 2 - Submitted?								
Timing Belt	Completed	92 Teeth, 460mm Length	1	$10.79	Free	3/4	Yes	https://www.amazon.com/q
Pnumatic Tubing	Completed	4mm OD/Blue	1	$10.99	Free	3/3	Yes	https://www.amazon.com/q
Turntable Bearing	Completed	NULL	1	$10.99	Free	3/3	Yes	https://www.amazon.com/q
Relay Module	Completed	NULL	1	$10.99	Free	03/03	Yes	https://www.amazon.com/q
ORDER 3 - Submitted								
Lead Screw	Completed	NULL	1	$9.99	Free	3/8	Yes	https://www.amazon.com/u
Linear Rod (8mm)	Completed	8mm x 100mm	1	$9.99	Free	3/8	Yes	https://www.amazon.com/d
Linear Bearing (8mm)	Completed	Null	1	$10.95	Free	3/8	Yes	https://www.amazon.com/L
ORDER 4 - Submitted								
Slip Ring	Completed	NULL	1	$14.99	Free	3/27	Yes	https://www.amazon.com/S
Bearings	Completed	15x21x4	1	$9.99	Free	3/27	Yes	https://www.amazon.com/q
Turntable Bearing	Completed	NULL	1	$10.99	Free	3/27	Yes	https://www.amazon.com/q
Solenoid Valve	Completed	NULL	1	$9.45	Free	3/27	Yes	https://www.amazon.com/4
ORDER 5 - Submitted								
Nema 17 Damper	Completed	NULL	1	$10.99	Free	3/30	Yes	https://www.amazon.com/E
M5x14mm Hex Bolt	Completed	50 pcs	1	$9.99	Free	3/30	Yes	https://www.amazon.com/1
M5 Hex Nut	Completed	NULL	1	$7.62	Free	3/30	Yes	https://www.amazon.com/S
Hall Effect Sensors	Completed	NULL	1	$11.52	Free	3/30	Yes	https://www.amazon.com/A
ORDER 6 - Submitted								
Stepper Motor (13Ncm)	Completed	17HS4023(13Ncm) / 1PCS	2	$19.98	Free	4/7	Yes	https://www.amazon.com/d
Stepper Motor (42Ncm)	Completed	17HS4401S(42N.cm) / 1 PC	1	$10.99	Free	4/7	Yes	https://www.amazon.com/d
Current Sensor	Completed	NULL	1	$7.99	Free	4/8	Yes	https://www.amazon.com/q
Addressable Led	Completed	NULL	1	$13.99	Free	4/7	Yes	https://www.amazon.com/B
ORDER 7 - Submitted								
M5 Tap & Bit	Completed	M5x.8 / M5x.8 Tap - 4.20mm	1	$7.01	Free	4/16	Yes	https://www.amazon.com/D
RFID - RDM6300	Completed	NULL	1	$7.79	Free	4/16	Yes	https://www.amazon.com/H
RFID - Tags	Completed	NULL	1	$8.99	Free	4/16	Yes	https://www.amazon.com/Y
M5 x 45mm Hex Bolt	Completed	M5 x 45mm / 50 pcs	1	$12.99	Free	4/17	Yes	https://www.amazon.com/q
Magnets	Completed	NULL	1	$12.99	Free	4/19	Yes	https://www.amazon.com/q
Wood Screws	Completed	#6 x 3/4"	1	$9.19	Free	4/17	Yes	https://www.amazon.com/W
Turntable Bearing	Completed	NULL	1	$10.99	Free	4/16	Yes	https://www.amazon.com/q
TOTALS			37	$481.33	$1.96			

V. Final Product

Recommended Final Product Information

In order to offer this product for sale to businesses there are several areas of the prototype that are in need of improvement. Additionally, there are legal issues that may need to be addressed before offering this product for sale

To begin, the center head unit responsible for the movement of medication should be changed to a snap fit assembly to prevent tampering and simplify construction. Additionally, when redesigning the central head units assembly it would be best to modify the routing of electrical wiring and pneumatic tubing to be run internally rather than externally.

Another major change to be made would be to make an enclosure for the entire unit to prevent tampering during operations. This shroud could either be a snap-fit cloche covering the entire unit, or simply walls with a door allowing access inside. To accompany this change, the electrical components currently located below the main head unit and pill bins should be placed into an enclosure to further prevent tampering and improve safety.

Some additional changes that can be made across the board relate to the materials that were used in prototyping. First off, all holding bins for the pills and the sorted pill bins are made of PLA which is not the best choice for mass production or recycling. For this reason the plastic materials that are having direct contact with the medication via storage should be made of polypropylene. Additionally, all of the custom pieces in the design made using an FDM 3D printer can be made using a different technique and material such as injection molding and ABS. Finally, the circular board used for the base does not need to be made of wood and could be made of a CNC cut and folded sheet metal or some plastic with the addition of struts to elevate the platform as is in the prototype.

Now for the gears and meshing plates that they operate on, those have been identified as one of the major slowdowns in the operation of the machine. It is believed that the use of 3D printed gears and meshing plates lead to excessive friction and skipping in the system, limiting the maximum speed and acceleration of the central head unit significantly. For this reason it is recommended that these parts be made of a much harder plastic than PLA, or better yet some aluminum alloy such as 6061. Regardless, if this change leads to a relatively significant increase in cost it would be the most beneficial as the machine would see drastic improvements in all facets of its performance.

Finally due to the nature of this product, the product must adhere to all legislation imposed by major governing bodies such as the DEA, EPA, and FDA. Some of the laws outlined by these organizations have already been addressed in the design by addressing the operating procedure, but others still need attention. An easily addressable issue is one concerning the dispensing of medication imposed by the DEA that requires all dispensed medication to have a label with the following information: fill date, pharmacy name/address, prescription serial number, patient and practitioner name, direction for use, and cautionary statements. This is a simple change that can be incorporated into the design through software by utilizing an existing printer at a pharmacy.

In the end, this product will need small changes to some of its key components before being offered for sale. Additionally, at the moment this product will need legal counsel before moving forward as there are too many potential barriers that could present themselves.

Packaging

For packaging the product for direct delivery to customers it is best that the majority of the pieces be left assembled, mainly the central sorting unit and the electronics box. To achieve this the product should be shipped in a single box composed of two to three smaller boxes inside. First off the outermost box will need to be roughly 500 mm x 500 mm x 500 mm. This size will allow for a buffer layer of packaging material between the additional boxes on the inside and the outside box preventing damage. The base, central sorting unit, pill bins, and sorted pill bin unit will each be placed in their own respective boxes. For the central sorting unit and the sorted pill bin unit bubble wrap should be wrapped around the product before being placed into their respective boxes as these parts are quite delicate. Alternatively, air pillows may be used to save on packaging cost and reduced waste. Lastly, once all individual boxes have been placed inside the main package some filler material should be placed inside such as loose fill peanuts for more air pillows. One final note to be made is that based on recommendation by SF Express, a logistics company based in Shenzhen China, the buffer material should have a thickness buffer of at least 5 cm given the final package would be roughly 5-10 kg. For a rough visual of how this would look see the figure below.

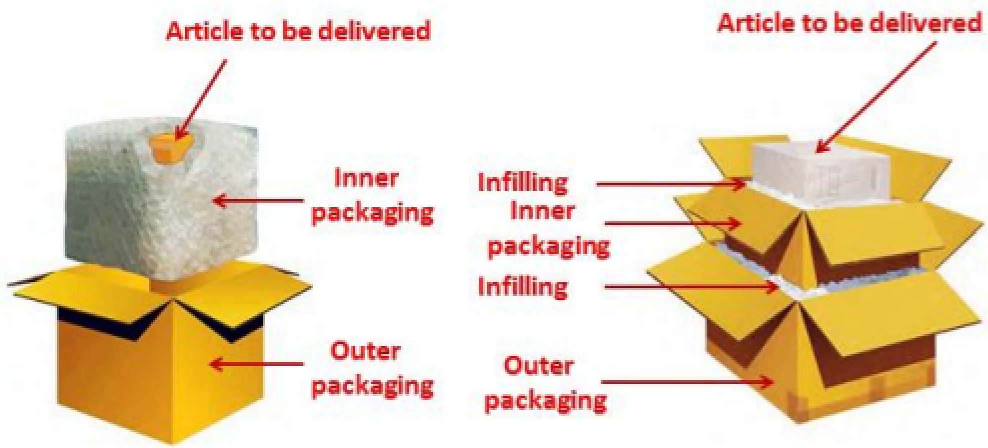


Figure 112: Packaging Concept

To further solidify how the product should be packaged and shipped the guidance provided by SF Express outlines the following for delicate or valuable items.

Packing for fragile items or valuable items
Outer packing: Double walled carton box or container with same strength
Inner packing: EPE sheet, cardboards
Cushioning: EPE sheet, bubble wrap, loose fill peanuts (EPE, EPS)

---

1. Wrap the contents separately with EPE sheet or bubble wrap. Then use cardboards or EPE sheet (thicker than 1cm) to separate the contents.

Used cardboards to protect the LED monitor before wrapping it with EPE sheet or bubble wrap.

2. Put the product in the center of the box to prevent it from touching the outer packing. Then fill up the space with sufficient cushioning material to avoid movement.

Figure 110: SF Express Guidelines

Manufacturing & Assembly Instructions

In the final product manufacturing would be reduced to two processes. First, the base board that all components are mounted to will need to be cut using some manner of a CNC machine. What type of CNC machine would depend on the material that is chosen for the final product's base. If wood is used for the base once again then a CNC router would suffice. In the case of a metal base a CNC plasma cutter or water jet could be used. The prototype shown in this paper was cut using a water jet (thanks to Frank Coffey) which is not ideal as it led to warping in the board, thus it is not recommended to use a waterjet on a wooden base. Second, all remaining components can be manufactured using polypropylene in an injection molding process. When manufacturing the pill storage bins it may be necessary to split the bins into two pieces that snap together to allow for the use of injection molding. Similarly, on some of the mounting plates a draft angle may need to be added to allow for the piece to release from the mold.

As was outlined in the previous section the product will be shipped in one large package with the individual pieces in separate boxes. By shipping the product his way it will allow for the delicate electronics to be assembled prior to the consumer receiving them. Once the package reaches the customer all that will need to be done by them is for the individual pieces to be mounted to the base plate. Due to the way that the components are designed each unique piece can only be attached in a single way. Additionally, all current mounting hardware is of the same size meaning the consumer need not worry about mixing bolts. In the end this results in consumers needing to only fasten 12 bolts and to snap the pill storage bins into place.

Cost Estimates

Located in the table below is the estimated cost for the machine's parts. Prices are derived from common consumer sites in order to calculate a reasonable cost estimation for individual builds. In some cases, individual costs were derived from larger quantity orders.

Table 29: Estimated Parts Cost

Parts				
Name	Amount	Type	Cost Per	Total Cost
Linear Rail	**2**	Part	$1.40	$2.80
Linear Bearing	**2**	Part	$0.51	$1.02
Lead Screw	**1**	Part	$2.16	$2.16
Lead Nut	**1**	Part	$0.83	$0.83
Juko Head	**1**	Part	$6.12	$6.12
Juko Quick Connect	**1**	Part	$10.50	$10.50
Pneumatic Connect (Small)	**1**	Part	$0.20	$0.20

Small Stepper	**1**	Part	$7.46	$7.46
Heat Insert	**1**	Part	$0.01	$0.01
M2 Machine Screw	**1**	Part	$0.02	$0.02
Stepper Damper	**3**	Part	$1.38	$4.14
M3 Bolt	**14**	Part	$0.03	$0.42
Optical Limit Switch	**1**	Part	$2.16	$2.16
Slip Ring	**1**	Part	$9.97	$9.97
Large Stepper	**2**	Part	$10.99	$21.98
Hall Limit Switch	**2**	Part	$1.69	$3.38
Vacuum Pump	**1**	Part	$14.13	$14.13
Solenoid Valve	**1**	Part	$6.80	$6.80
Relay Bank	**1**	Part	$6.23	$6.23
CNC Shield	**1**	Part	$2.49	$2.49
Stepper Driver	**3**	Part	$1.04	$3.12
Arduino Uno	**2**	Part	$6.25	$12.50
Power Supply	**1**	Part	$12.46	$12.46
Power Socket	**1**	Part	$0.92	$0.92
Power Cord	**1**	Part	$3.09	$3.09
Pneumatic Connect (Large)	**2**	Part	$0.25	$0.50
Pneumatic Tube (Large)	**1**	Part	$1.15	$1.15
Pneumatic Tube (Small)	**2**	Part	$1.00	$2.00
M5 Nut	**20**	Part	$0.08	$1.60
Base Board	**1**	Part	$11.94	$11.94
Usb Cable	**2**	Part	$3.60	$7.20
Pneumatic Dual Connect	**1**	Part	$0.14	$0.14
Led Light String (23 count)	**1**	Part	$5.87	$5.87
RFID Board	**1**	Part	$2.20	$2.20

RFID Coil	**1**	Part	$0.60	$0.60
RFID Tag	**1**	Part	$0.72	$0.72
Long M5 Bolt	**18**	Part	$0.90	$16.20
Short M5 Bolt	**20**	Part	$0.60	$12.00
Wood Screw	**46**	Part	$0.04	$1.84
				198.87

Additionally, a fixed $40 filament cost should be included to the estimate to allow for all 3d parts to be printed and accommodate for any failed prints.

VI. ABET Outcomes

At the heart of this project has always been the goal of improving the lives of those that struggle with adhering to pharmaceutical drug routines by addressing the problem directly. This problem is not limited to any demographic of the population, but is more likely to occur in those that must manage many medications, and the elderly. Unfortunately, these two groups tend to coincide creating a problem that is much more dangerous for the patient.. Due to the potential health risks by not appropriately adhering to medication guidelines the duty of managing medications often will fall on to a caregiver after the patient has made mistakes in their medication routine. This is the outcome that this project seeks to prevent by providing medication to the patient pre-sorted and ready for consumption. In preventing poor pharmaceutical drug adherence this project is able to address major issues in public and social environments.

In terms of addressing the effect that this project has on public health the benefits should be quite obvious. By providing the patient all of their medication directly, already sorted, the risk of improper sorting is completely eliminated. Additionally, a common problem in drug adherence is having to make multiple trips to the pharmacy, so by giving a patient all their medication at once this may also be avoided. In a medication nonadherence overview by Frost & Sullivan, it was found that “...only 20-30% of medical prescriptions are filled by patients”[1]. The reason given for this lack of prescription filling is people not wanting to make a trip to the pharmacy multiple times in a month. This is something, as previously mentioned, that this project has been able to easily address. Another common issue that leads to poor medication adherence has to do with medication label literacy. This issue is not limited strictly to the patients and their familial caregivers but also to professional caregivers. In a study by Lindquist, Jain, Tam, Martin, and Baker titled *Inadequate Health Literacy Among Paid Caregivers of Seniors,* they found “Inadequate health literacy was found in 35.7% of caregivers; 60.2% of all caregivers made errors with the pillbox test medications, showing difficulty in following label directions.”[2]. These numbers illustrate that even of the professional caregivers available for hire over a third of them have health literacy issues and nearly two thirds of them make errors when sorting medication. This truly shows the need to address this problem before the patient, and as shown the caregiver, prior to either of them touching the medication in order to prevent improper dosing. In another study by Grindrod, Gates, Dolovich, et al, they looked at using a smartphone app called ClereMed to identify adults' medication literacy based on their ability to properly read medication labels. In their finding that write, “ClereMed correctly identified 72% (5/7) of participants with functional reading difficulty, and 63% (5/8) who failed a real-life pill-sorting task, but only 21% (6/28) of participants with cognitive impairment.”[3]. This study shows quite well that the vast majority of patients are unable to both interpret their medication labels as well as properly sort the medication. The interesting bit here is that of those in this study only about ⅕ had some cognitive impairment. This means that even without some cognitive impairment most people are incapable of managing their medications. As previously stated, the issues with medication literacy are completely avoided through the device proposed in this report. By providing the medication to the patient all at once and sorted, the patient or the caregiver need not have the appropriate health literacy to perform the task.

Now, moving on to the impacts that this project will have on society, primarily in terms of jobs. In hospital and hospice settings it is standard procedure for health care staff to handle the sorting and distribution of medication for patients. This process often takes a significant amount of time and is quite error prone. There have been many unique ideas to try and speed up the process as well as offload some of the work to patients. One of these projects is called Dr. Fill and was created by a team of researchers at the Department of Medical Informatics in Aachen, Germany which seeks to gamify the routine. The idea the team has is to create a physical game board that interprets if the patient sorts the pills correctly and then awards a score to the patient which goes to a leaderboard for the care facility [4]. Now this seems like a good idea but why not simply implement the solution in this report of having a machine due the sorting. There is no benefit to the medical facility in gamifying the process and it does not properly address the issue at hand, as outlined in the previous paragraph. Seeking to gamify the problem will simply lead to more headaches for the healthcare staff and potentially more work. By implementing a sorting machine to address this issue the time spent sorting medications for patients can be reallocated to other tasks. It would also be likely that a reduction in hours for the medical staff could be seen as they no longer need to sort medications by hand. This idea of the potential loss of hours or even jobs is one that scares healthcare workers, and rightfully so, but it is not what should be expected with the introduction of automated medicine. In the paper *The Impact of Robotics and Automation on Working Conditions and Employment* the team writes of how automation in this age will be collaborative in nature writing,

> “ This augmented collaborative workforce is the wave of the future and has enormous implications for employment in the automation age. It will redefine the relations between workers, their crafts, and their working environments. On the one hand, workers can focus on aspects that require creativity, social skills, and emotional intelligence; on the other, this could also have a dehumanizing effect if workers’ activities are subjugated to robots’ behaviors.”[5].

This quote clearly states all of the benefits and potential pitfalls that can occur for those who have to adapt to new technologies in the workplace. In terms of this idea applied to the medical industry, this should be seen as a huge boon, mainly due to the single phrase "...workers can focus on aspects that require creativity, social skills, and emotional intelligence…". For patients they often do not get the attention that they require from medical staff due to the large volume of patients compared to medical staff. This is why providing automation anywhere applicable in the healthcare sector should be pursued as it will not only improve the patients physical health but also their mental health. Obviously there are the potential drawbacks that this could have on the healthcare workers but this should be quite limited as the healthcare industry is quite hands on when compared to industrial sectors, which is primarily what the quote above is referencing. At its core the healthcare industry is concerned with improving the health of the patient and maintaining and supporting its healthcare professionals which are both accomplished through the implementation of automated pill sorting.

References

1. "Personalization in Medical Dispensing." TechVision Opportunity Engines, 12 Aug. 2016. Frost & Sullivan, Pill Dispension. Accessed 21 Mar. 2022.

2. Lindquist, Lee A et al. "Inadequate health literacy among paid caregivers of seniors." Journal of general internal medicine vol. 26,5 (2011): 474-9. doi:10.1007/s11606-010-1596-2

3. Grindrod, Kelly Anne et al. "ClereMed: Lessons Learned From a Pilot Study of a Mobile Screening Tool to Identify and Support Adults Who Have Difficulty With Medication Labels." JMIR mHealth and uHealth vol. 2,3 e35. 15 Aug. 2014, doi:10.2196/mhealth.3250

4. Bukowski, Mark et al. "Gamification of Clinical Routine: The Dr. Fill Approach." Studies in health technology and informatics vol. 225 (2016): 262-6.

5. Pham, Q.-C., et al. "The Impact of Robotics and Automation on Working Conditions and Employment [Ethical, Legal, and Societal Issues]." IEEE Robotics & Automation Magazine, vol. 25, no. 2, June 2018, pp. 126–128., https://doi.org/10.1109/mra.2018.2822058.

Appendix 1 - Order of Magnitude Estimations

1. **Operating Costs**

Assumed values - Pharmacy:
Pharmacy Tech Hourly Rate - 15 $/hr
Tech Fill Rate - 20 scripts/hr
Techs on Staff - 3
Daily Operating Hours - 12 hrs
Actual Time Spent Working - 11 hrs

Calculations - Pharmacy:
Labor = Pharmacy Tech Hourly Rate * Tech Fill Rate
Labor = 15*20 = 0.75 $/script

Combined Hourly Tech Rate = Pharmacy Tech Hourly Rate * Techs on Staff
Combined Hourly Tech Rate = 15*3 = 45 $/hr

Total Fill Rate = Tech Fill Rate * Techs on Staff
Total Fill Rate = 20*3 = 60 scripts/hr

Total Labor Costs = Daily Operating Hours * Combined Hourly Tech Rate
Total Labor Costs = 45*12 = $540

Assumed Values - Machine:
Machine Cost: $500
Power Consumption: 50W
Electricity Cost: $ 0.07/kWh

Calculations - Machine
Scripts to Break Even = Machine Cost / Labor
Script to Break Even = 500 / 0.75 = 667 scripts

Scripts to Match Techs = Total Fill Rate * Daily Operating Hours
Scripts to Match Techs = 60 * 12 = 720 scripts

Operating Cost = Power Consumption * Electricity Cost * Daily Operating Hours
Operating Cost = 50/1000 * 0.07 * 12 = 0.042 $/day
= 0.042 * 7 = 0.294 $/week

2. **Forces**

Assumed Values:
Max Pill Weight: 5000 mg
Coefficient of Static Friction: 0.3
Smallest Pill Diameter: 1.5mm
Ultimate Strength: 27.578 MPa
Atmospheric Pressure: 101 kPa
Suction Diameter: 1mm
Max Pill Mass: 5000 mg

Calculations:

Max Gravitational Force = Max Pill Mass * 9.81
Max Gravitational Force = 5000 mg * 9.81 = 0.049 N

Smallest Area = $\pi/4*d^2$
Smallest Area = $\pi/4*1.5^2$ = 1.767 mm^2

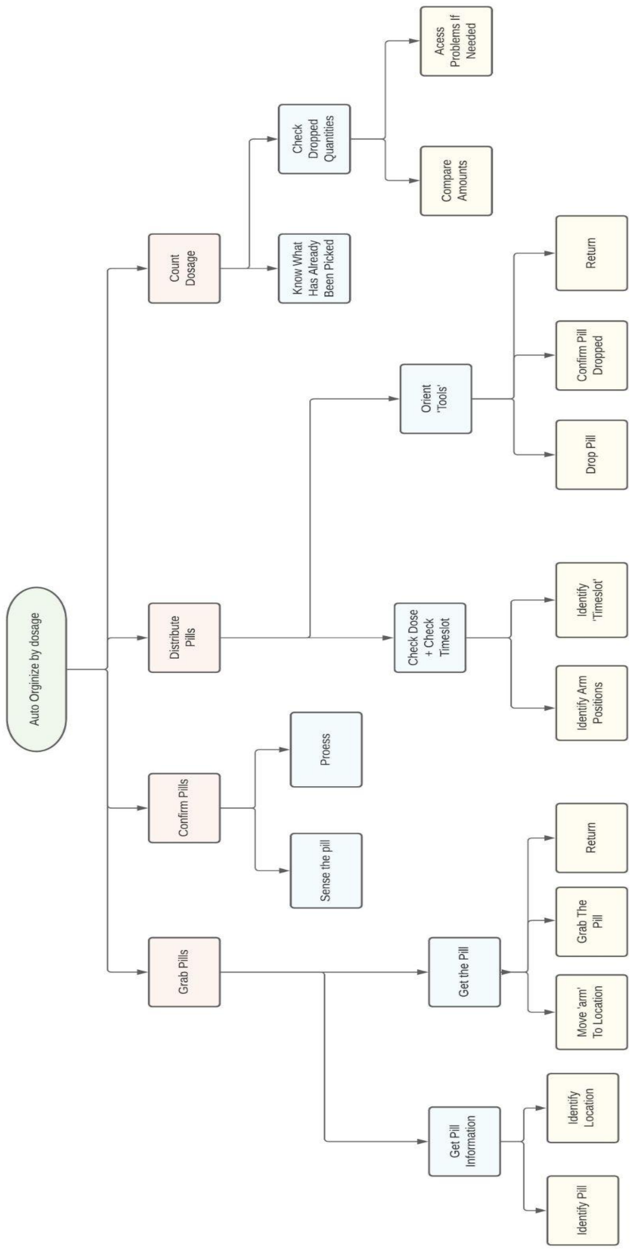
Force to Grip = ½ * Max Gravitational Force / Coefficient of Static Friction
Force to Grip = 0.5 * 0.049 * 0.3 = 0.082 N

Minimum Force to Break = Ultimate Strength * Smallest Area
Minimum Force to Break = 27.578 * 1.767 = 48.734 N

Suction Area = $\pi/4*d^2$
Suction Area = $\pi/4*1^2$ = 0.785 mm^2

Force to Break Suction = Atmospheric Pressure * Suction Area
Force to Break Suction = 101 * 0.785 = 0.079

Appendix 2 - Function Chart



Appendix 3 - Conceptual Design Calculations

Concept #2:

Pill Box Sizing
Pill Bottle Diameter = 40 mm
Pill Bottle Height = 60 mm
Volume of Pill Bottle = $\pi r^2 * h = 75000 mm^3$

000 Pills
- Mass = 1641 g
- Length = 26.1 mm
- Length Cap = 12.95 mm
- Volume of Cap = $\pi r^2 * h + 4/3 \pi r^3 = 562.69 mm^3$

Assumed Pill Box of 40 mm cube
Volume of Cube = $l * w * h = 64000 mm^3$
Max Pills = Volume of Cube / Volume of Cap = 113.739 capsules

Sorted Pill Box Sizing
Number of Pills = 15
Volume per Division = Number of Pills * Volume of Cap = 15 * 562.69 = 8440.35 mm^3
Total Volume = Day of Week * Volume per Division = 7 * 8440.35 = 59082.45 mm^3
Assumed Box Height = 20 mm
Sorted Pill Box Diameter = (Total Volume * 4 / π / h) ^½ = 61.3 mm

SMT Range
Arc Length = 200 mm
Angle = π / 2
Inner Radius = Arc Length / Angle = 127.3239 mm
Box Depth = 40 mm
Outer Radius = Inner Radius + Box Depth = 167.3239 mm

Appendix 4 - Embodiment Design Calculations

SMT Unit Stepper:
J = 1/12 * ρ * A * B * C * (A^2 + B^2)
J = 1/12 * 2 * 13.6 * 1.8 * 1.925 * (13.6^2 + 1.8^2) g-cm^2
J = 6651.5525

Rotor Inertia = J // Ratio
Rotor Inertia = 6651.5525 //[10,5,1]
Rotor Inertia = [665,1330,6651] g-cm^2

Sorted Pill Stepper:

$J = \frac{1}{8} * m * (D^2/4 + L^{\frac{2}{3}})$
$J = \frac{1}{8} * 1415 * (9.1^2/4 + 49/3)$
$J = 6550.713$ g-cm^2

Rotor Inertia = J // Ratio
Rotor Inertia = 6550.713 // [10,5,1]
Rotor Inertia = [665,1310,6651]

Appendix 5 - Software Code

Python

Main.py

```
from Pill_Sorting_Interface import Pill_Sorting_Interface

def main():
   # Hardcoded database
   user_info = [{"id": "1234",
                 "first_name": 'John',
                 "last_name": "Smith",
                 "age": "72",
                 "address": "96 Whooovile RD KT",
                 "prescription": {"Advil": 2, "Motrin": 3},
                 "frequency": [1, 1],
                 "days_of_week": [[1, 0, 1, 0, 0, 0, 0], [1, 0, 1, 0, 1, 0,
0]]
                 }, {"id": "1281",
                     "first_name": 'Jane',
                     "last_name": "Doe",
                     "age": "65",
                     "address": "12 Seuss Valley MO",
                     "prescription": {"Benadryl": 2, "Motrin": 6},
                     "frequency": [2, 2],
                     "days_of_week": [[1, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 1,
0, 1]]
                     },{"id": "1",
                     "first_name": 'Test1',
                     "last_name": "Test1",
                     "age": "999",
                     "address": "Test1",
                     "prescription": {"Aspirin": 14},
                     "frequency": [2],
                     "days_of_week": [[1, 1, 1, 1, 1, 1, 1]]
                     },{"id": "2",
                     "first_name": 'Test2',
```

```
                        "last_name": "Test2",
                        "age": "999",
                        "address": "Test2",
                        "prescription": {"Benadryl": 4, "Motrin": 6},
                        "frequency": [2, 2],
                        "days_of_week": [[1, 0, 0, 1, 0, 0, 0], [0, 1, 0, 0, 1,
0, 1]]
                        },{"id": "3",
                        "first_name": 'Test3',
                        "last_name": "Test3",
                        "age": "999",
                        "address": "Test3",
                        "prescription": {"Benadryl": 2, "Motrin": 6,"Aspirin":
2, "Advil": 6,"Tylenol": 2, "Claritin": 6},
                        "frequency": [2, 2, 2, 2, 2, 2],
                        "days_of_week": [[1, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 1,
0, 1],[1, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 1, 0, 1],[1, 0, 0, 0, 0, 0, 0],
[0, 1, 0, 0, 1, 0, 1]]
                        },{"id": "4",
                        "first_name": 'Test4',
                        "last_name": "Test4",
                        "age": "999",
                        "address": "Test4",
                        "prescription": {"Benadryl": 2, "Motrin": 6,"Aspirin":
2, "Advil": 6,"Tylenol": 2, "Claritin": 6},
                        "frequency": [2, 2, 2, 2, 2, 2],
                        "days_of_week": [[1, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 1,
0, 1],[1, 0, 0, 0, 0, 0, 0], [0, 1, 0, 0, 1, 0, 1],[1, 0, 0, 0, 0, 0, 0],
[0, 1, 0, 0, 1, 0, 1]]
                        }]
   Pill_Sorting_Interface(user_info)


if __name__ == "__main__":
   main()
```

Pill Sorting Interface

```
import configparser

import serial
import serial.tools.list_ports
from PyQt5.QtCore import QSortFilterProxyModel, Qt
from PyQt5.QtGui import QIcon, QStandardItemModel, QStandardItem
from PyQt5.QtWidgets import QApplication, QMenuBar, QWidget, QGridLayout,
QGroupBox, QLineEdit, QHBoxLayout, QLabel, \
   QTableView, QHeaderView, QVBoxLayout, QTextEdit, QPushButton, QMessageBox,
QCheckBox
```

```python
from Sorting_Timed_Message import SortingMessageBox
from Sorting_Pill_Dialog import Sorting_Pill_Dialog
from Configuration_Interface import Configuration_Interface
from Direct_Control_Interface import Direct_Control_Interface


class Pill_Sorting_Interface():
   def __init__(self, scripts):
       self.app = QApplication([])
       self.scripts = scripts
       self.ser_grbl = None
       self.ser_uno = None
       self.configurator = None
       self.controller = None
       self.sorter = None
       self.current_selection = None
       self.current_selection_string = None
       self.gcode = None

       self.config = "config.ini"

       self.menu_bar = QMenuBar()
       self.set_com_ports()
       if self.ser_grbl is not None and self.ser_uno is not None:

           self.create_table()
           self.create_top_right_window()
           self.create_bottom_right_window()
           # self.create_progress_bar()
           self.create_menu_bar()

           self.window = QWidget()

           main_layout = QGridLayout()
           main_layout.addWidget(self.menu_bar, 0, 0)
           main_layout.addWidget(self.left_window, 1, 0, 4, 3)
           main_layout.addWidget(self.top_right_window, 1, 3)
           main_layout.addWidget(self.bottom_right_window, 2, 3)
           # main_layout.addWidget(self.progress_bar, 3, 3)
           main_layout.setRowMinimumHeight(0, 25)
           main_layout.setColumnMinimumWidth(1, 750)
           main_layout.setColumnMinimumWidth(3, 250)
           main_layout.setRowMinimumHeight(1, 500)
           main_layout.setColumnStretch(3, 1)
           main_layout.setColumnStretch(0, 2)

           self.window.setWindowTitle("Pill Sorting Interface")
           self.window.setLayout(main_layout)
```

```
            self.window.setWindowIcon(QIcon('Pills-icon.png'))
            self.window.show()
        else:
            self.serial_not_set()

        self.app.exec()

    # Create table and table search field
    def create_table(self):
        self.left_window = QGroupBox("Open Prescriptions")

        self.model = QStandardItemModel(len(self.scripts), len(self.scripts[0])
- 3)
        self.model.setHorizontalHeaderLabels(["ID", "First Name", "Last Name",
"Age", "Address"])

        for i, user in enumerate(self.scripts):
            for j, key in enumerate(user.keys()):
                if key != "prescription" and key != "frequency" and key !=
"days_of_week":
                    self.model.setItem(i, j, QStandardItem(user[key]))

        filter_model = QSortFilterProxyModel()
        filter_model.setSourceModel(self.model)
        filter_model.setFilterCaseSensitivity(Qt.CaseInsensitive)
        filter_model.setFilterKeyColumn(2)

        search_bar = QLineEdit()
        search_bar.setStyleSheet('font-size: 15px')
        search_bar.setPlaceholderText("Search by last name")
        layout = QHBoxLayout()
        label = QLabel("Search")
        layout.addWidget(label)
        layout.addWidget(search_bar)
        search_bar.textChanged.connect(filter_model.setFilterRegExp)

        self.table = QTableView()
        self.table.setStyleSheet('font-size: 15px')

        self.table.setModel(filter_model)

self.table.selectionModel().selectionChanged.connect(self.update_information)
        self.table.horizontalHeader().setSectionResizeMode(QHeaderView.Stretch)

        layout = QVBoxLayout()
        layout.addWidget(search_bar)
        layout.addWidget(self.table)
```

```python
        self.left_window.setLayout(layout)

    # Creates prescription information field
    def create_top_right_window(self):
        self.top_right_window = QTextEdit()
        self.top_right_window.setStyleSheet('font-size: 15px')
        self.top_right_window.setPlainText("Prescription Information")
        self.top_right_window.setReadOnly(True)

    # Creates start,pause,abort button field
    def create_bottom_right_window(self):
        self.bottom_right_window = QGroupBox("Controls")

        self.scan_rfid_button = QPushButton("Scan RFID")
        self.scan_rfid_button.clicked.connect(self.scan_rfid)

        self.start_button = QPushButton("Start")
        self.start_button.clicked.connect(self.start_sort)

        layout = QGridLayout()
        layout.addWidget(self.scan_rfid_button, 0, 0, 1, 3)
        layout.addWidget(self.start_button, 1, 0, 1, 3)

        self.bottom_right_window.setLayout(layout)

    # Creates menu bar
    def create_menu_bar(self):
        self.menu_bar = QMenuBar()

        tools = self.menu_bar.addMenu("Tools")
        tools.addAction("Configuration", self.start_configurator)
        tools.addAction("Direct Control", self.start_direct_control)

    # Starts configuration tool
    def start_configurator(self):
        self.configurator = Configuration_Interface(self.ser_grbl)
        self.configurator.show()


    # Starts the direct control tool
    def start_direct_control(self):
        self.controller = Direct_Control_Interface(self.ser_grbl)
        self.controller.show()


    def start_sorter(self,steps = False):
        self.sorter = Sorting_Pill_Dialog(self.ser_grbl, self.ser_uno,
self.current_selection['prescription'],
```

```
                                              self.gcode,steps)
        self.sorter.show()


    # Set the serial communication attribute
    def set_com_ports(self):
        ports = serial.tools.list_ports.comports()
        for i, port in enumerate(ports):
            print(port.description)
            if port.description.startswith("Arduino"): # tty for linux
                self.ser_uno = serial.Serial(port=port.device, baudrate=115200,
timeout=1)
            elif port.description.startswith("USB"):
                self.ser_grbl = serial.Serial(port=port.device,
baudrate=115200, timeout=1)
                self.ser_grbl.write('$X\n'.encode())

    # Display messagebox that serial ports are not connected
    def serial_not_set(self):
        msg = QMessageBox()
        msg.setWindowTitle("Missing Serial Port")
        msg.setText("Please connect both USBs")
        msg.setInformativeText(
            "If both USBs are connected and you are still getting this error
please try different ports")
        msg.setIcon(QMessageBox.Critical)

        msg.exec_()

    # Scan rfid sticker
    def scan_rfid(self):
        self.ser_uno.write(b'scan')
        rfid_info = self.ser_uno.readline()

    # Begin sorting process
    def start_sort(self):
        if self.current_selection is not None:
            self.ser_uno.write(b'start')

            check_box = QCheckBox("Step by Step?")

            msg = QMessageBox()
            msg.setWindowTitle("Load Medication")
            msg.setText("Please load medication")
            msg.setInformativeText(self.current_selection_string)
            msg.setIcon(QMessageBox.Information)
            msg.setCheckBox(check_box)
```

```
            msg.setStandardButtons(QMessageBox.Ok | QMessageBox.Cancel)

            response = msg.exec_()
            inSteps = check_box.isChecked()

            if response == QMessageBox.Ok:
                self.generate_gcode()
                starting_msg = SortingMessageBox(timeout=5)
                starting_msg.exec_()
                self.start_sorter(inSteps)
        else:
            msg = QMessageBox()
            msg.setWindowTitle("No Prescription Selected")
            msg.setText("Please select a prescription from the table or use an 
RFID tag")
            msg.setIcon(QMessageBox.Critical)

            msg.exec_()

    # Updates the prescriptions information field
    def update_information(self, selected, deselected):
        try:
            item = selected.indexes()[0]
            self.current_selection = self.scripts[item.row()]

            script_info = self.scripts[item.row()]["prescription"]
            script_string = "Slot".ljust(15) + "Medication".ljust(20) + "\n"
            script_string += "-" * 30 + "\n"
            i = 1
            for key, val in script_info.items():
                script_string += f"Slot {i}".ljust(15) + f"{key}:
{val}".ljust(20) + "\n"
                i += 1

            self.current_selection_string = script_string
            self.top_right_window.setPlainText(script_string)
        except IndexError:
            print("IndexError")

    # Generate the gcode commands for sorting
    def generate_gcode(self):
        config = configparser.ConfigParser()
        config.read(self.config)

        home = config['POSITIONS']['Home']
        safe = config['POSITIONS']['Safe']
        p1_x = float(config['POSITIONS']['P1'].split(" ")[0][1:])
        p2_x = float(config['POSITIONS']['P2'].split(" ")[0][1:])
```

```
        p3_y = float(config['POSITIONS']['P3'].split(" ")[1][1:])
        p4_y = float(config['POSITIONS']['P4'].split(" ")[1][1:])

        x_gap = p2_x - p1_x
        y_gap = p4_y - p3_y

        self.gcode = []
        self.gcode.append('$X\n')
        self.gcode.append(f'G90 {safe}\n')



        # Sort a single medication type at a time
        current_slot = p1_x
        for i, val in
enumerate(self.current_selection['prescription'].values()):

            # Get frequency and days of the week
            freq = self.current_selection['frequency'][i]
            days = [j for j, v in
enumerate(self.current_selection['days_of_week'][i]) if v]

            for day in days:
                # Rotate sorted pill bin to appropriate day
                self.gcode.append(f'G90 Y{day*y_gap}\n')  # TODO this will need
tweaking

                # Drop freq number of pills in the corresponding day
                for _ in range(freq):
                    self.gcode.append(f'G90 X{current_slot}\n')  # move to slot
                    self.gcode.append(f'M8\n')  # spindle on
                    self.gcode.append('G90 Z-5\n')  # descend
                    self.gcode.append('G90 Z0\n')  # ascend
                    self.gcode.append(f'G90 {home}\n')  # return to drop
                    self.gcode.append(f'M9\n')  # spindle off

            current_slot += x_gap
```

Configuration Interface

```
from PyQt5.QtCore import Qt
from PyQt5.QtGui import QIcon
from PyQt5.QtWidgets import QWidget, QGridLayout, QGroupBox, QLabel,
QDoubleSpinBox, QPushButton, QMessageBox


class Configuration_Interface(QWidget):
   def __init__(self, ser):
       super().__init__()
```

```
        self.ser = ser
        layout = QGridLayout()

        self.create_fields()
        layout.addWidget(self.settings_group, 0, 0)

        self.setWindowTitle("Configuration")
        self.setWindowIcon(QIcon("Gear-icon"))
        self.setLayout(layout)
        self.setWindowModality(Qt.ApplicationModal)
        self.setAttribute(Qt.WA_DeleteOnClose)

    def create_fields(self):
        fields = ["Step Pulse", "Step Idle Delay", "Step Port Invert Mask",
"Dir Port Invert Mask",
                  "Step Enable Invert", "Limit Pins Invert",
                  "Probe Pin Invert", "Status Report Mask", "Junction
Deviation", "Arc Tolerance", "Report Inches",
                  "Soft Limits", "Hard Limits",
                  "Homing Cycle", "Homing Dir Invert Mask", "Homing Feed",
"Homing Seek", "Homing Debounce",
                  "Homing Pull-Off", "X Step/mm", "Y Step/mm", "Z Step/mm",
                  "X Max Rate", "Y Max Rate", "Z Max Rate", "X Accel", "Y
Accel", "Z Accel", "X Max Travel",
                  "Y Max Travel", "Z Max Travel"]

        self.settings_group = QGroupBox("Settings")
        settings = self.get_current_settings()

        self.setting_boxs = []
        layout = QGridLayout()
        i = 0
        j = 0
        for field, setting in zip(fields, settings):
            command, starting_value = setting

            label = QLabel(field)
            value_box = QDoubleSpinBox(minimum=-2000, maximum=2000)
            value_box.setValue(starting_value)

            self.setting_boxs.append([field, command, starting_value,
value_box])

            if j % 2 == 0:
                layout.addWidget(label, i, 0)
                layout.addWidget(value_box, i, 1, 1, 2)
            else:
                layout.addWidget(label, i, 3)
```

```
                layout.addWidget(value_box, i, 4, 1, 2)
                i += 1

            j += 1

        apply_button = QPushButton("Apply")
        apply_button.clicked.connect(self.update_settings)
        layout.addWidget(apply_button, i + 1, 0, 1, 6)
        self.settings_group.setLayout(layout)

    def update_settings(self):
        changes = []
        for i, box in enumerate(self.setting_boxs):
            field, command, starting_value, spin_box = box

            current_value = spin_box.value()
            if current_value != starting_value:
                print(f'{command}={current_value}\n')
                self.ser.write(f'{command}={current_value}\n'.encode())
                changes.append(f"({field}):\t {starting_value} --->
{current_value}")
                self.setting_boxs[i][2] = current_value

        alert = QMessageBox()
        alert.setWindowTitle("Changes Applied")

        if changes:
            alert.setText("Changes")
            alert.setInformativeText("\n".join(changes))
        else:
            alert.setText("No Changes Made")

        alert.exec_()

    def get_current_settings(self):
        # Clear serial
        while self.ser.readline().decode():
            pass

        # Get current settings
        self.ser.write('$$\n'.encode())
        responses = []
        text = self.ser.readline().decode()
        while text:
            responses.append(text)
            text = self.ser.readline().decode()

        # Extract settings
```

```
        values = []
        for response in responses:
            if "=" in response:
                response = response.split("=")
                command = response[0]
                value = float(response[1].split(" ")[0])
                values.append([command, value])

        return values

    def closeEvent(self, event):
        event.accept()
```

Direct Control Interface

```
import configparser
import time
from functools import partial

from PyQt5.QtCore import Qt, QThread
from PyQt5.QtGui import QIcon
from PyQt5.QtWidgets import QWidget, QGridLayout, QGroupBox, QPushButton,
QLabel, QSpinBox, QFrame, QPlainTextEdit, \
   QVBoxLayout, QMessageBox

from Worker_Threads import Serial_Reader


class Direct_Control_Interface(QWidget):
    def __init__(self, ser=None):
        super().__init__()
        layout = QGridLayout()

        self.left_group = None
        self.serial_input_box = None
        self.serial_output_box = None
        self.ser = ser
        self.config = configparser.ConfigParser()
        self.config.read("config.ini")

        self.create_top_button_group()
        self.create_serial_input()
        self.create_serial_output()
        self.create_positioning_group()

        layout.addWidget(self.positioning_group, 0, 0, 4, 1)
        layout.addWidget(self.left_group, 0, 4, 2, 2)
        layout.addWidget(self.serial_input_box, 2, 4, 2, 2)
        layout.addWidget(self.serial_output_box, 0, 6, 4, 1)
```

```
        self.setWindowTitle("Direct Control")
        self.setWindowIcon(QIcon("Gear-icon"))
        self.setLayout(layout)
        self.setWindowModality(Qt.ApplicationModal)
        self.setAttribute(Qt.WA_DeleteOnClose)

    def create_top_button_group(self):
        self.left_group = QGroupBox("Controls")
        layout = QGridLayout()

        # X Control Buttons
        plus_x10 = QPushButton("X +10")
        plus_x10.clicked.connect(partial(self.move_axis_relative, 'x', 10))
        minus_x10 = QPushButton("X -10")
        minus_x10.clicked.connect(partial(self.move_axis_relative, 'x', -10))

        plus_x1 = QPushButton("X +1")
        plus_x1.clicked.connect(partial(self.move_axis_relative, 'x', 1))
        minus_x1 = QPushButton("X -1")
        minus_x1.clicked.connect(partial(self.move_axis_relative, 'x', -1))

        plus_x001 = QPushButton("X +0.01")
        plus_x001.clicked.connect(partial(self.move_axis_relative, 'x', 0.01))
        minus_x001 = QPushButton("X -0.01")
        minus_x001.clicked.connect(partial(self.move_axis_relative, 'x',
-0.01))

        # Y Control Buttons
        plus_y10 = QPushButton("Y +10")
        plus_y10.clicked.connect(partial(self.move_axis_relative, 'y', 10))
        minus_y10 = QPushButton("Y -10")
        minus_y10.clicked.connect(partial(self.move_axis_relative, 'y', -10))

        plus_y1 = QPushButton("Y +1")
        plus_y1.clicked.connect(partial(self.move_axis_relative, 'y', 1))
        minus_y1 = QPushButton("Y -1")
        minus_y1.clicked.connect(partial(self.move_axis_relative, 'y', -1))

        plus_y001 = QPushButton("Y +0.01")
        plus_y001.clicked.connect(partial(self.move_axis_relative, 'y', 0.01))
        minus_y001 = QPushButton("Y -0.01")
        minus_y001.clicked.connect(partial(self.move_axis_relative, 'y',
-0.01))

        # Z Control Buttons
        plus_z10 = QPushButton("Z +10")
        plus_z10.clicked.connect(partial(self.move_axis_relative, 'z', 10))
```

```
        minus_z10 = QPushButton("Z -10")
        minus_z10.clicked.connect(partial(self.move_axis_relative, 'z', -10))

        plus_z1 = QPushButton("Z +1")
        plus_z1.clicked.connect(partial(self.move_axis_relative, 'z', 1))
        minus_z1 = QPushButton("Z -1")
        minus_z1.clicked.connect(partial(self.move_axis_relative, 'z', -1))

        plus_z001 = QPushButton("Z +0.01")
        plus_z001.clicked.connect(partial(self.move_axis_relative, 'z', 0.01))
        minus_z001 = QPushButton("Z -0.01")
        minus_z001.clicked.connect(partial(self.move_axis_relative, 'z',
-0.01))

        # User input control
        x_spinner_label = QLabel("X")
        x_spinner = QSpinBox(minimum=0, maximum=1000)

        y_spinner_label = QLabel("Y")
        y_spinner = QSpinBox(minimum=0, maximum=1000)

        z_spinner_label = QLabel("Z")
        z_spinner = QSpinBox(minimum=0, maximum=1000)

        send_button = QPushButton("Send")
        send_button.clicked.connect(partial(self.move_absolute, x_spinner,
y_spinner, z_spinner))

        horizontal_separator = QFrame()
        horizontal_separator.setGeometry(60, 110, 751, 20)
        horizontal_separator.setFrameShape(QFrame.HLine)
        horizontal_separator.setFrameShadow(QFrame.Sunken)

        vertical_separator1 = QFrame()
        vertical_separator1.setGeometry(60, 110, 751, 20)
        vertical_separator1.setFrameShape(QFrame.VLine)
        vertical_separator1.setFrameShadow(QFrame.Sunken)

        vertical_separator2 = QFrame()
        vertical_separator2.setGeometry(60, 110, 751, 20)
        vertical_separator2.setFrameShape(QFrame.VLine)
        vertical_separator2.setFrameShadow(QFrame.Sunken)

        # Group Buttons
        # Independent axis control buttons
        layout.addWidget(plus_x10, 0, 0, 1, 4)
        layout.addWidget(minus_x10, 0, 4, 1, 4)
        layout.addWidget(plus_x1, 1, 0, 1, 4)
```

```
        layout.addWidget(minus_x1, 1, 4, 1, 4)
        layout.addWidget(plus_x001, 2, 0, 1, 4)
        layout.addWidget(minus_x001, 2, 4, 1, 4)

        layout.addWidget(vertical_separator1, 0, 9, 3, 1)

        layout.addWidget(plus_y10, 0, 10, 1, 4)
        layout.addWidget(minus_y10, 0, 14, 1, 4)
        layout.addWidget(plus_y1, 1, 10, 1, 4)
        layout.addWidget(minus_y1, 1, 14, 1, 4)
        layout.addWidget(plus_y001, 2, 10, 1, 4)
        layout.addWidget(minus_y001, 2, 14, 1, 4)

        layout.addWidget(vertical_separator2, 0, 19, 3, 1)

        layout.addWidget(plus_z10, 0, 20, 1, 4)
        layout.addWidget(minus_z10, 0, 24, 1, 4)
        layout.addWidget(plus_z1, 1, 20, 1, 4)
        layout.addWidget(minus_z1, 1, 24, 1, 4)
        layout.addWidget(plus_z001, 2, 20, 1, 4)
        layout.addWidget(minus_z001, 2, 24, 1, 4)

        layout.addWidget(horizontal_separator, 4, 0, 1, 28)

        layout.addWidget(x_spinner_label, 5, 0)
        layout.addWidget(x_spinner, 5, 1, 1, 6)
        layout.addWidget(y_spinner_label, 5, 7)
        layout.addWidget(y_spinner, 5, 8, 1, 6)
        layout.addWidget(z_spinner_label, 5, 14)
        layout.addWidget(z_spinner, 5, 15, 1, 6)
        layout.addWidget(send_button, 5, 21, 1, 7)

        self.left_group.setLayout(layout)

    def create_serial_input(self):
        self.serial_input_box = QGroupBox("Serial Input")

        self.serial_input_field = QPlainTextEdit()
        self.serial_input_field.setStyleSheet('font-size: 15px')

        send_button = QPushButton("Send")
        send_button.clicked.connect(self.write_serial)
        clear_button = QPushButton("Clear")
        clear_button.clicked.connect(self.serial_input_field.clear)
        help_button = QPushButton("Help")
        help_button.clicked.connect(self.help_button_info)

        layout = QGridLayout()
```

```
        layout.addWidget(self.serial_input_field, 0, 0, 3, 8)
        layout.addWidget(help_button, 3, 0, 1, 2)
        layout.addWidget(clear_button, 3, 2, 1, 2)
        layout.addWidget(send_button, 3, 4, 1, 4)

        self.serial_input_box.setLayout(layout)

    def create_serial_output(self):
        self.serial_output_box = QGroupBox("Serial Output")
        self.serial_output_field = QPlainTextEdit()
        self.serial_output_field.setReadOnly(True)

        clear_button = QPushButton("Clear")
        clear_button.clicked.connect(self.serial_output_field.clear)

        layout = QVBoxLayout()
        layout.addWidget(self.serial_output_field)
        layout.addWidget(clear_button)
        self.serial_output_box.setLayout(layout)

        self.thread = QThread()
        self.worker = Serial_Reader(self.ser)
        self.worker.moveToThread(self.thread)
        self.thread.started.connect(self.worker.update_display)
        self.worker.progress.connect(self.serial_output_field.appendPlainText)
        self.worker.update.connect(self.update)
        self.thread.start()

    def create_positioning_group(self):
        self.positioning_group = QGroupBox("Configure Positions")
        layout = QGridLayout()

        set_home_button = QPushButton("Set Home")
        set_home_button.clicked.connect(partial(self.set_pos, 0))
        move_button = QPushButton("Move Home")
        move_button.clicked.connect(partial(self.move_to_preset, 0))
        layout.addWidget(set_home_button, 0, 0, 1, 4)
        layout.addWidget(move_button, 0, 4, 1, 4)

        for i in range(0, 12, 2):
            set_text = ""
            move_text = ""
            pos = None

            if i // 2 == 0:
                set_text = "Set Home"
                move_text = "Move Home"
                pos = "Home"
```

```
        elif i // 2 == 1:
            set_text = "Set Safe"
            move_text = "Move Safe"
            pos = "Safe"
        else:
            set_text = f"Set P{i // 2 - 1}"
            move_text = f"Move P{i // 2 - 1}"
            pos = f"P{i // 2 - 1}"

        set_button = QPushButton(set_text)
        set_button.clicked.connect(partial(self.set_pos, pos))
        move_button = QPushButton(move_text)
        move_button.clicked.connect(partial(self.move_to_preset, pos))

        layout.addWidget(set_button, i, 0, 1, 4)
        layout.addWidget(move_button, i, 4, 1, 4)
        if i != 10:
            horizontal_separator = QFrame()
            horizontal_separator.setGeometry(60, 110, 751, 20)
            horizontal_separator.setFrameShape(QFrame.HLine)
            horizontal_separator.setFrameShadow(QFrame.Sunken)
            layout.addWidget(horizontal_separator, i + 1, 0, 1, 8)

    self.positioning_group.setLayout(layout)

def move_to_preset(self, pos):
    move = self.config['POSITIONS'][pos]
    print(move)
    self.ser.write(f'G90 {move}\n'.encode())

def set_pos(self, pos):
    self.worker.set_allowed(False)
    time.sleep(1)
    self.ser.write('?\n'.encode())

    position = self.ser.readline().decode('ascii')
    print(position)

    self.worker.set_allowed(True)

    position = position.split(",")

    x_coord = float(position[1].split(":")[1])
    y_coord = float(position[2])
    z_coord = float(position[3])

    gcode = f"X{x_coord} Y{y_coord} Z{z_coord}"
```

```
        self.config.set("POSITIONS", pos, gcode)

        with open('config.ini', 'w') as configfile:
            self.config.write(configfile)

    def move_axis_relative(self, axis, dist):
        self.ser.write(f'G91 {axis}{dist}\n'.encode())

    def move_absolute(self, x, y, z):
        self.ser.write(f'G90 X{x.value()} Y{y.value()}
Z{z.value()}\n'.encode())

    def help_button_info(self):
        msg = QMessageBox()
        msg.setWindowTitle("Additional Information")
        msg.setText("Commands:")
        msg.setInformativeText("Enter the $ character for the Grbl help
command\n")

        msg.exec_()

    def write_serial(self):
        commands = self.serial_input_field.toPlainText().encode()
        self.ser.write(commands)
        self.serial_input_field.clear()

    def closeEvent(self, event):
        self.thread.terminate()
        event.accept()
```

Sorting Pill Dialog

```
from PyQt5.QtCore import Qt, QThread
from PyQt5.QtGui import QIcon
from PyQt5.QtWidgets import QWidget, QLabel, QGridLayout, QPushButton,
QGroupBox, QPlainTextEdit, QVBoxLayout, \
    QMessageBox, QProgressBar

from Worker_Threads import Sorting_Worker
from time import time


class Sorting_Pill_Dialog(QWidget):
    def __init__(self, ser_grbl, ser_uno, pills, gcode, inSteps):
        super().__init__()
        print(gcode)
        self.ser_grbl = ser_grbl
        self.ser_uno = ser_uno
        self.pills = pills
```

```
        self.gcode = gcode
        self.inSteps = inSteps

        self.thread_response = False

        self.sorting = True

        self.resume_pause = None
        self.abort = None
        self.progress_bar = None
        self.description_field = None

        self.text = QLabel(f"Sorting Pills")

        self.create_buttons()
        self.create_progress_bar()
        self.start_thread()

        layout = QGridLayout()

        layout.addWidget(self.text, 0, 0)
        layout.addWidget(self.resume_pause, 1, 0, 2, 2)
        layout.addWidget(self.abort, 1, 2, 2, 2)
        layout.addWidget(self.progress_bar, 3, 0, 1, 4)

        self.setWindowTitle("Sorting Pills")
        self.setWindowIcon(QIcon("Pills-icon"))
        self.setLayout(layout)
        self.setWindowModality(Qt.ApplicationModal)
        self.setAttribute(Qt.WA_DeleteOnClose)

        self.start_time = time()

    # Create resume/pause and abort buttons
    def create_buttons(self):
        self.resume_pause = QPushButton("Pause")
        self.resume_pause.clicked.connect(self.change_state)

        self.abort = QPushButton("Abort")
        self.abort.clicked.connect(self.abort_sort)

    # Create sorting worker thread
    def start_thread(self):
        self.thread = QThread()
        self.worker = Sorting_Worker(self.ser_grbl, self.ser_uno,self.pills,
self.gcode, self.inSteps, self)
        self.worker.moveToThread(self.thread)
        self.thread.started.connect(self.worker.sort_and_update)
```

```
        self.worker.progress.connect(self.update_progress_bar)
        self.worker.stepper.connect(self.next_step_box)
        self.worker.finished.connect(self.final_dialog)
        self.thread.start()

    def next_step_box(self):
        msg = QMessageBox()
        msg.setWindowTitle("Proceed or Rerun")
        msg.setText("Proceed to next step or rerun previous cycle")
        msg.setIcon(QMessageBox.Information)
        msg.setStandardButtons(QMessageBox.Yes | QMessageBox.Cancel)

        proceed = msg.button(QMessageBox.Yes)
        proceed.setText("Proceed")

        rerun = msg.button(QMessageBox.Cancel)
        rerun.setText("Rerun")

        msg.exec_()

        if msg.clickedButton() == proceed:
            self.worker.increment_pos(1)
        else:
            self.worker.decrement_pos(5)
            self.update_progress_bar(self.progress_bar.value() - 5)


        self.thread_response = True

    # Notify of successful sort
    def final_dialog(self):
        self.thread.terminate()
        final_time = time()

        msg = QMessageBox()
        msg.setWindowTitle("Sorting Complete")
        msg.setIcon(QMessageBox.Information)
        msg.setText(f"Sorting has been completed\n Time: {round(final_time -
self.start_time,3)} seconds")

        msg.exec_()

        self.close()

    # Creates progress bar
    def create_progress_bar(self):
        self.progress_bar = QProgressBar()
        self.progress_bar.setRange(0, len(self.gcode)-1)
```

```
        self.progress_bar.setValue(0)

    def update_progress_bar(self,n):
        self.progress_bar.setValue(n)

    # Change running state when button is pressed
    def change_state(self):
        if self.sorting:
            self.setWindowTitle("Sorting Paused")
            self.resume_pause.setText("Resume")
            self.sorting = False
            self.worker.set_sorting(False)
            self.ser_uno.write(b'hold')

        else:
            self.setWindowTitle("Sorting Pills")
            self.resume_pause.setText("Pause")
            self.sorting = True
            self.worker.set_sorting(True)
            self.ser_uno.write(b'resume')

    # Abort sorting process
    def abort_sort(self):
        self.sorting = False
        self.ser_uno.write(b'abort')
        self.close()

    # Close window
    def closeEvent(self, event):
        self.progress_bar.setValue(0)
        event.accept()
```

Worker Threads

```
from PyQt5.QtCore import pyqtSignal, QObject
from time import sleep,time
import re


class Serial_Reader(QObject):
    progress = pyqtSignal(str)
    update = pyqtSignal()


    def __init__(self, serial_port):
        super().__init__()
        self.ser = serial_port
        self._allowed = True

    def update_display(self):
```

```
        if self.ser is not None:

            while True:
                while self._allowed:
                    text = self.ser.readline().decode('ascii')
                    if text:
                        self.progress.emit(text)
                        self.update.emit()

    def set_allowed(self, b):
        self._allowed = b


class Sorting_Worker(QObject):
    progress = pyqtSignal(int)
    stepper = pyqtSignal()
    finished = pyqtSignal()


    def __init__(self, serial_grbl, serial_uno ,pills, gcode, inSteps,
mainapp):
        super().__init__()
        self._sorting = True
        self.ser = serial_grbl
        self.ser_uno = serial_uno
        self.pills = pills
        self.gcode = gcode
        self.inSteps = inSteps

        self.app = mainapp

        self.i = 0

        self.pill_counts = list(self.pills.values())


    def sort_and_update(self):
        self.i = 0
        instructions_completed = 0
        pattern = re.compile("^[^YZM]*$")

        current_x = None
        pos = 0
        self.ser.write(self.gcode[self.i].encode())
        self.i += 1

        while True:
            if self._sorting:
```

```
                if self.i < len(self.gcode):
                    self.ser.write(self.gcode[self.i].encode())
                    next_x = pattern.match(self.gcode[self.i])

                    if self.gcode[self.i].startswith("M9"):
                        if self.inSteps:
                            self.stepper.emit()

                            # Await messagebox to be clicked by main thread
                            while not self.app.thread_response:
                                sleep(0.1)

                            self.app.thread_response = False

                    elif next_x is not None:
                        if current_x is None:
                            current_x = self.gcode[self.i]
                            self.ser_uno.write(f"P{pos}".encode())
                        else:
                            if current_x != self.gcode[self.i]:
                                current_x = self.gcode[self.i]
                                pos += 1
                                self.ser_uno.write(f"P{pos}".encode())




                    self.i += 1

                while self.ser.readline().decode('ascii').strip() != 'ok':
                    pass

                instructions_completed += 1
                self.progress.emit(instructions_completed)


            if instructions_completed == len(self.gcode):
                break


        self.ser_uno.write("done".encode())
        self.finished.emit()


    def set_sorting(self, state):
        self._sorting = state

    def decrement_pos(self,j):
```

```
        self.i -= j

    def increment_pos(self,j):
        self.i += j
```

Sorting Timed Message

```
from PyQt5.QtCore import QTimer
from PyQt5.QtWidgets import QMessageBox


class SortingMessageBox(QMessageBox):
    def __init__(self, timeout=3, parent=None):
        super(SortingMessageBox, self).__init__(parent)

        self.setWindowTitle("Starting Sort")
        self.setIcon(QMessageBox.Information)
        self.setStandardButtons(QMessageBox.NoButton)
        self.time_to_wait = timeout
        self.timer = QTimer(self)
        self.timer.setInterval(1000)
        self.timer.timeout.connect(self.change_time)
        self.change_time()
        self.timer.start()

    def change_time(self):
        self.setText(f"Sorting will begin in {self.time_to_wait} seconds")
        if self.time_to_wait <= 0:
            self.close()
        self.time_to_wait -= 1

    def closeEvent(self, event):
        self.timer.stop()
        event.accept()
```

Config.ini

```
[POSITIONS]
home = X0.0 Y0.0 Z0.0
safe = X0.0 Y0.0 Z0.0
p1 = X-0.536 Y0.0 Z0.0
p2 = X-0.428 Y0.0 Z0.0
p3 = X0.864 Y0.0 Z0.0
p4 = X0.864 Y1.0 Z0.0
```

Arduino

```
#include <Adafruit_NeoPixel.h>
```

```
#define PIN 3
#define LEDS 23

#define valvePin 11
#define holdPin 12
#define abortPin 13

// LED Strip
Adafruit_NeoPixel strip = Adafruit_NeoPixel(LEDS,PIN,NEO_GRB + NEO_KHZ800);

// Variable to store information from serial
String incomingString;
int lightPos;
int blinkCount;




void setup() {
 Serial.begin(115200);

 strip.begin();
 strip.show();

 pinMode(holdPin, OUTPUT);
 pinMode(abortPin, OUTPUT);
 pinMode(valvePin, OUTPUT);
}

void loop() {
 if (Serial.available() > 0){
   incomingString = Serial.readString();

   // If moving to a new position turn corresponding light on
   if (incomingString.startsWith("P")){
       strip.clear();
       lightPos = incomingString.substring(1).toInt();
       strip.setPixelColor(lightPos,0,0,255);
       strip.show();
     }

   // Handle lights for start sequence
   else if(incomingString.equals("start")){
       strip.clear();
       strip.fill(strip.Color(255,0,0),0,LEDS);
       strip.show();
     }
```

```
    // Handle lights for paused sequence
    else if(incomingString.equals("hold")){
        strip.clear();
        strip.fill(strip.Color(255,255,0),0,LEDS);
        strip.show();

        digitalWrite(holdPin,HIGH);
      }

    else if(incomingString.equals("resume")){
        strip.clear();
        strip.fill(strip.Color(0,255,0),0,LEDS);
        strip.show();
        digitalWrite(holdPin,LOW);
      }

    // Handle lights for abort
    else if(incomingString.equals("abort")){
        strip.clear();
        strip.fill(strip.Color(0,255,0),0,LEDS);
        strip.show();

        digitalWrite(abortPin,HIGH);
      }

    else if(incomingString.equals("done")){
      blinkCount = 0;
      while(blinkCount < 5){
          strip.clear();
          strip.rainbow(0,1,255,255,true);
          strip.show();
          delay(1000);
          strip.clear();
          strip.show();
          delay(250);
          blinkCount += 1;
        }
      }
    else if(incomingString.equals("open")){
      digitalWrite(valvePin,HIGH);
      }
    else if(incomingString.equals("close")){
      digitalWrite(valvePin,LOW);
      }

    // Handle rfid scan
    else if(incomingString.equals("scan")){
        Serial.print("scanning");
```

```
        }
    }

}
```

Appendix 6 - Patents

In conducting patent research there are many patents that have been granted but expired for pill sorting and dispensing devices. Additionally, many of the existing devices are meant to be used for rapid pill counting to expedite the prescription filling process for pharmacists. The types of solutions are not what this project is considered of so only patents granted for automated pill sorting and dispensing will be examined at this time.

In the realm of pill sorting most available patents have been granted for machines that are used for rapidly filling individual pill bottles with a single type of medication such as this device. This device accepts labeled bins of pills and then puts them into prescription bottles and attaches appropriate labels. This patent was granted on 2011-10-04 but has since expired.

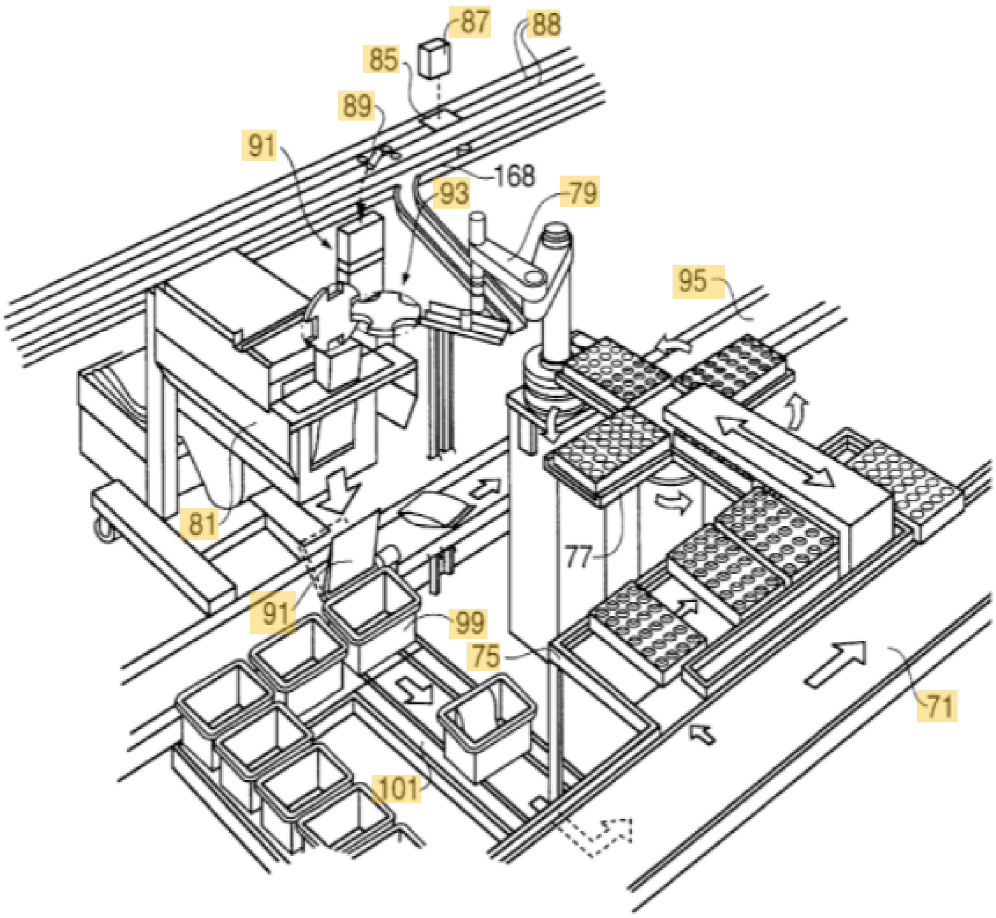


Figure: Automatic prescription filling, sorting and packaging system [1]

Another such device that was designed for use in pharmacies is this device which actually goes about sorting mixed medications. The machine does so by taking partial images of the individual pills and comparing them to reference images and then placing them into their appropriate bins. This machine only seeks to sort medication into bins not fill prescriptions. For this design the patent was granted in 2020-04-28 and is still active.

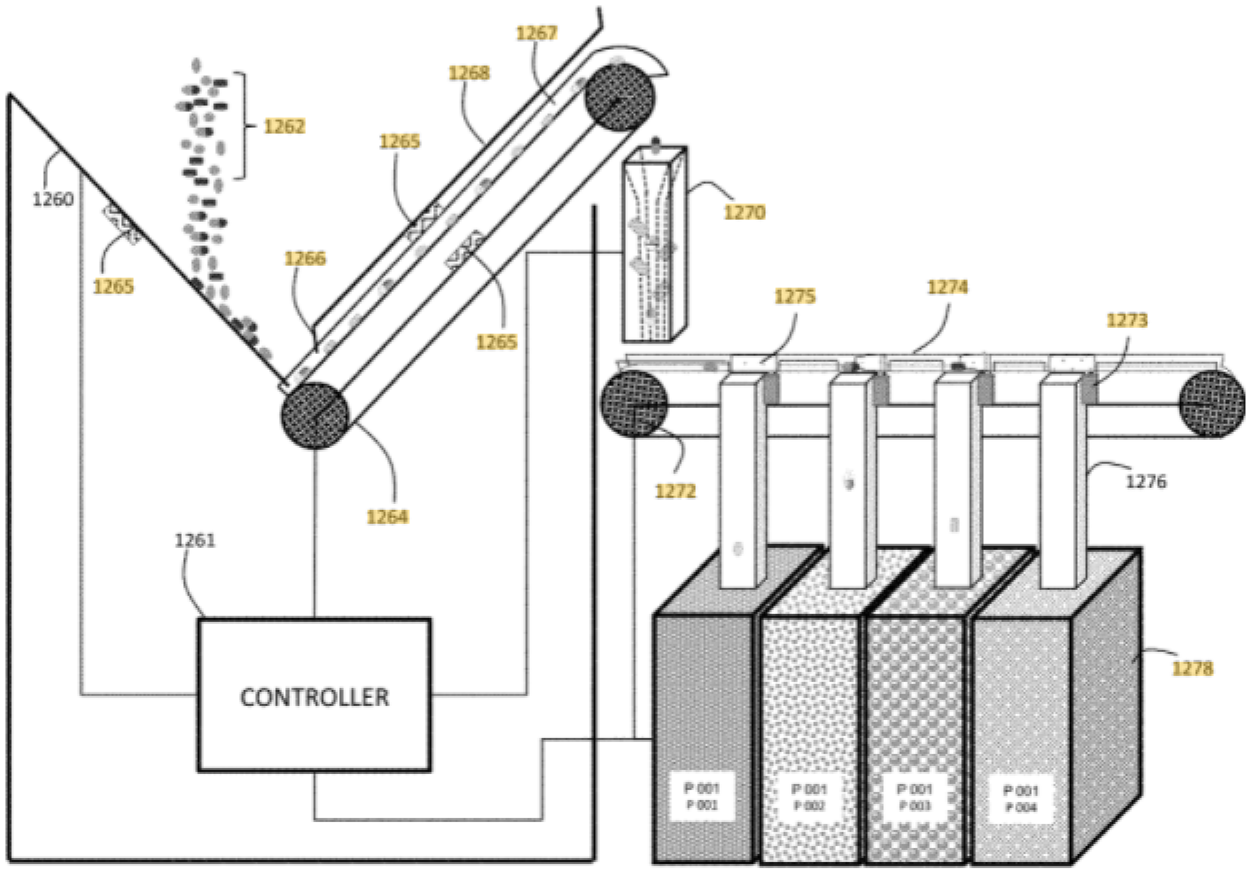


Figure: Method and device for identification and/or sorting of medicines [2]

Another device that a patent has been granted for is this device which temporarily stores pills into a rotating drum with pill bins located below on a conveyor belt. The machine keeps a predetermined type and number of pills in the rotating drum's segments. When a desired pill is to be dispensed the conveyor belt moves the pill bottle into place below the outlet and the drum rotates into the correct position and the pill is dispensed. In addition the device includes a counter to verify the correct number of pills have been dispensed. This patent was granted on 2008-07-8 and is still active.

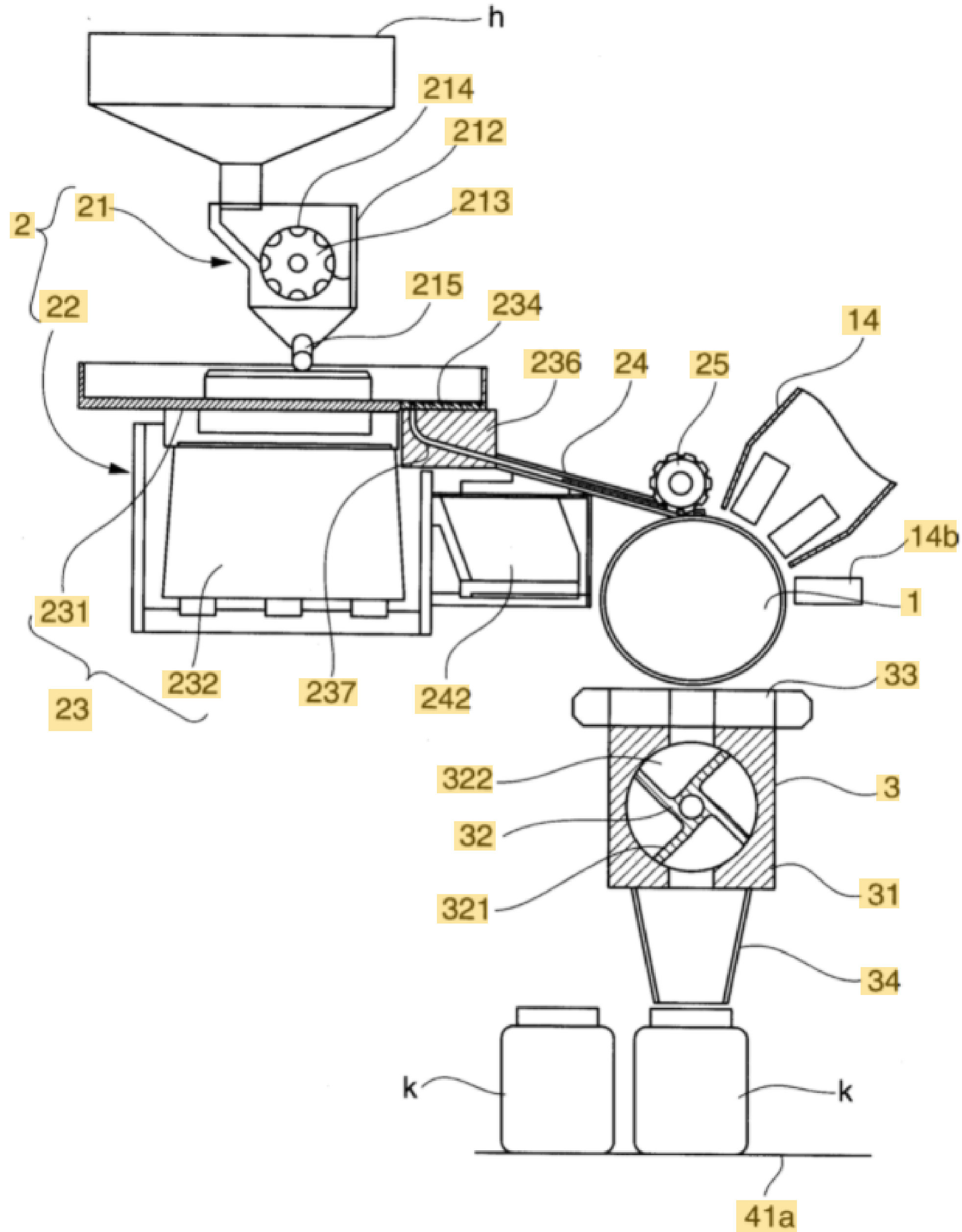


Figure: Counting and feeding device for small article [3]

In this next device a singular vacuum head is utilized along with specialized storage bins that allow for a single article to be extracted at a time. The system consists of many bins and a singular head that navigates from bin to bin to fill individual prescriptions. This patent was granted on 2012-01-10 and has since expired.

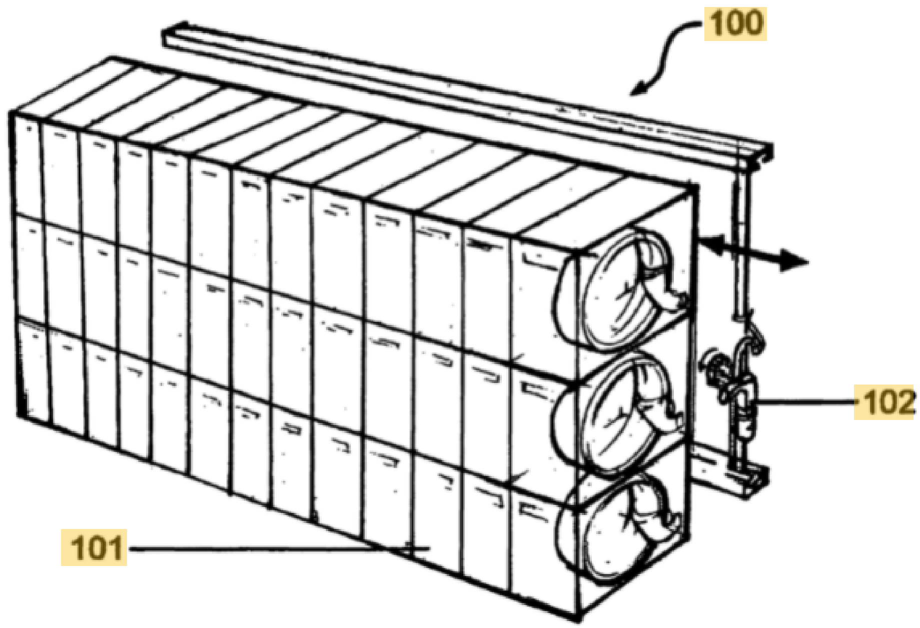


Figure: Automated article dispensation mechanism [4]

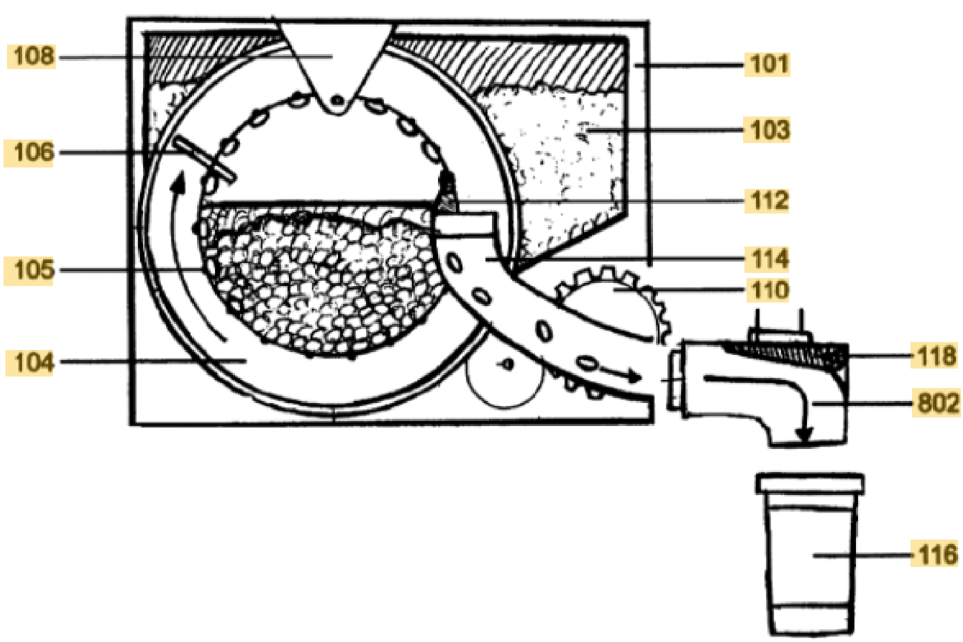


Figure: Automated article dispensation mechanism [4]

Another device meant for automated pill dispensing is this device which does so using an opening and closing mechanism that engages with a hopper of pills and counts the output. The main advantage of this device is that the entire machine is contained within a chassis making it difficult for unauthorized individuals to tamper with the medication or the device. This patent was originally submitted in 2004 and has since been abandoned.

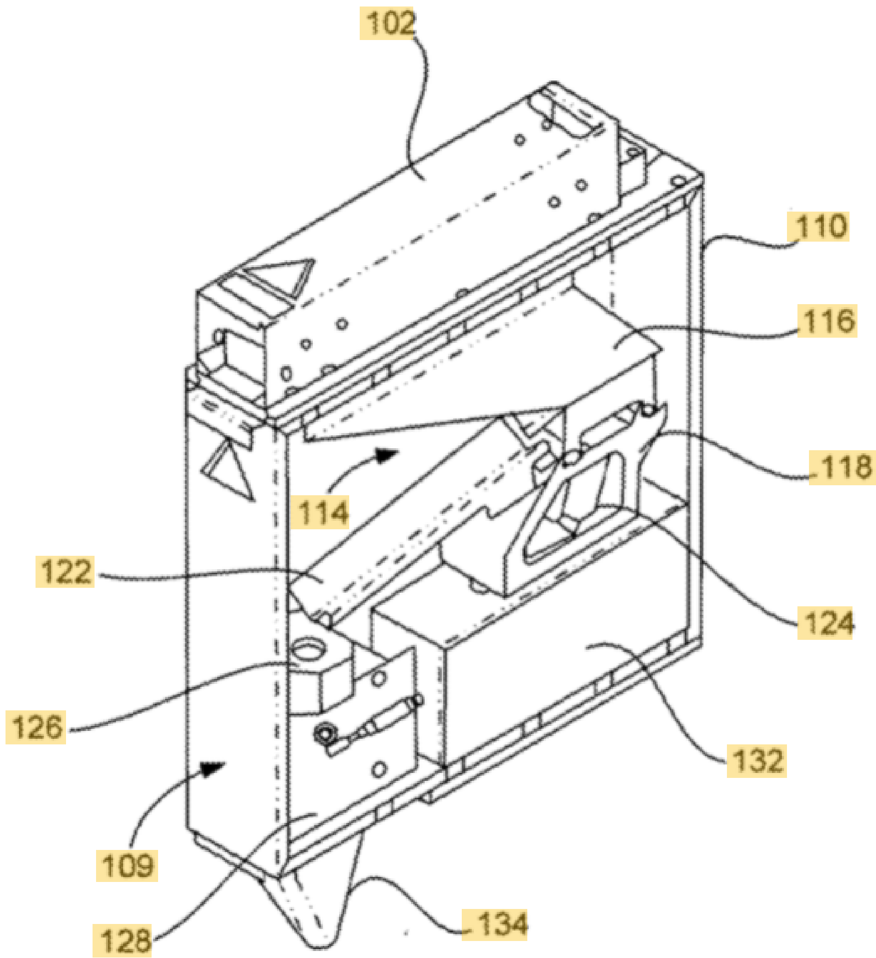


Figure: Systems and Methods of Automated Tablet Dispensing, Prescription Filling, and Packaging [5]

One unique device that was granted a patent was this device which uses individual spinning drums to house reservoirs of individual types of pills. When a pill is needed for dispensing the drum rotates and a pill is grabbed by an air blower. Once the pills have settled in the drum the blower dispenses the pill down a chute which goes into the pill container. This patent was granted on 2008-06-04 and has since expired.

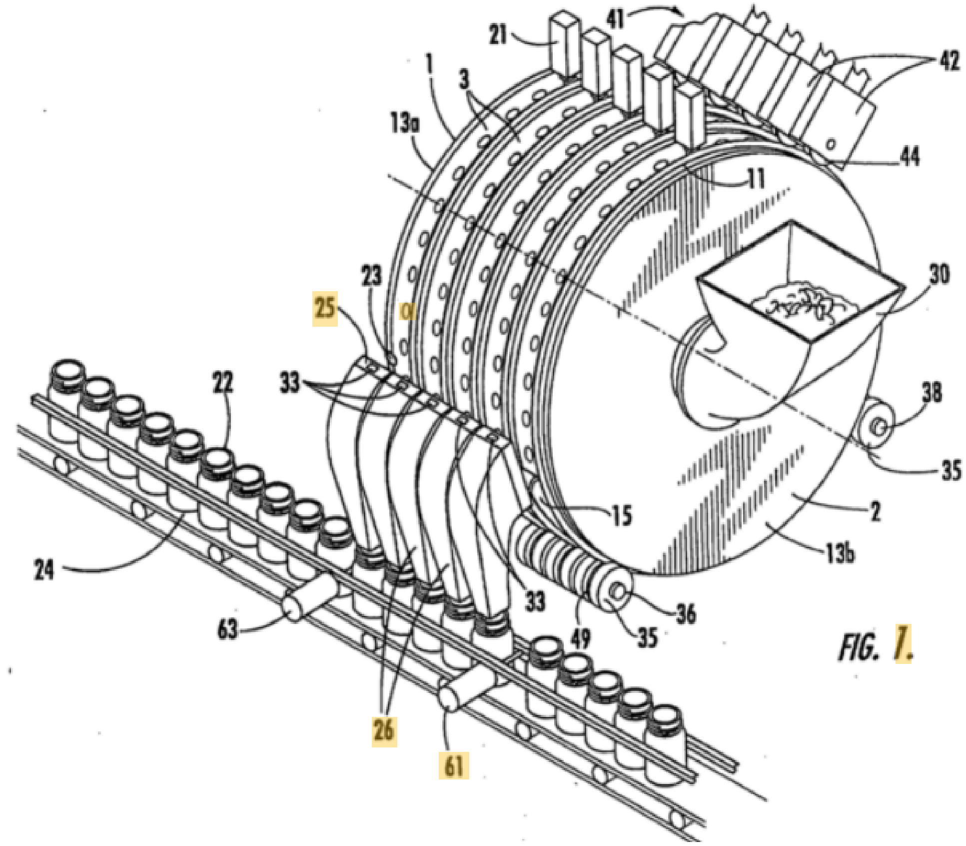


Figure: Apparatus and methods for filling containers with pills [6]

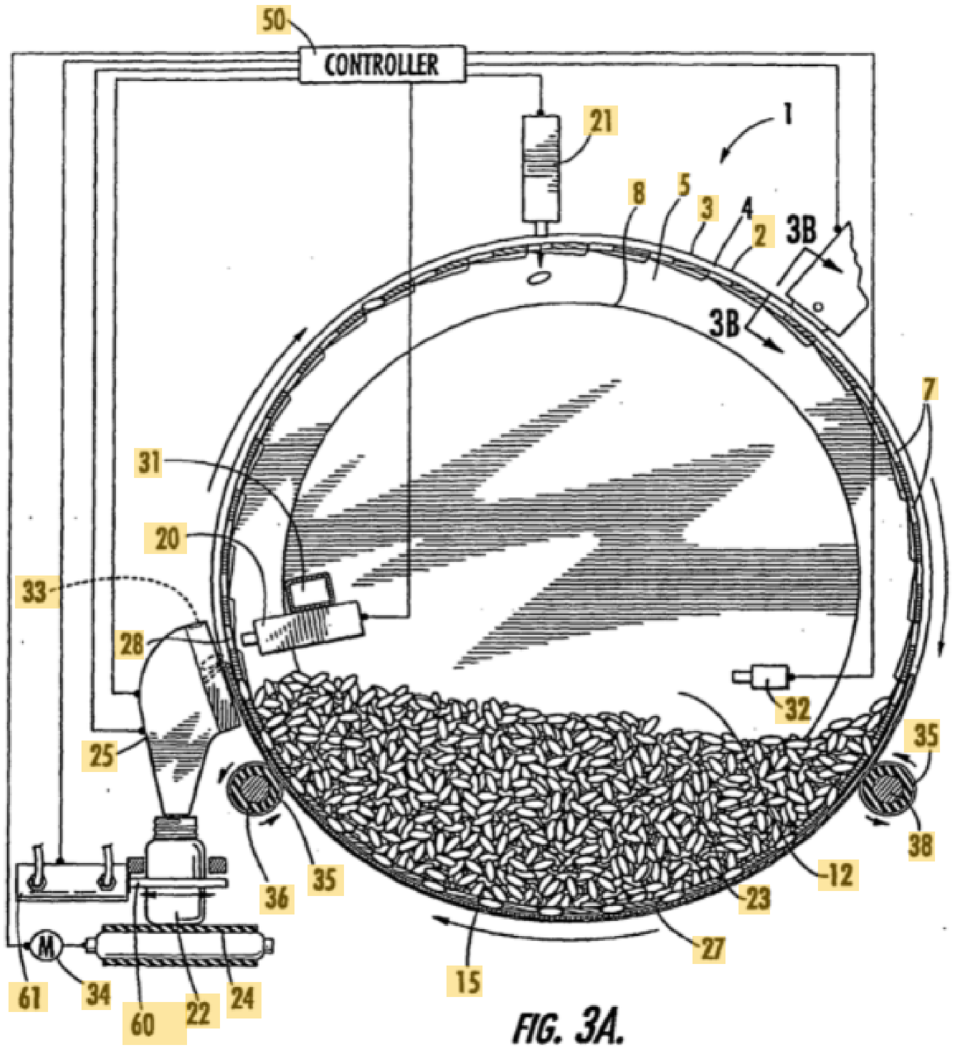


Figure: Apparatus and methods for filling containers with pills [6]

Another device is this one which uses a v-channel guide and a plunger to guide pills single-file into correct bin locations. The main use of this device is to dispense the correct number of pills into their correct locations. This patent was granted on 1987-06-30 has since expired.

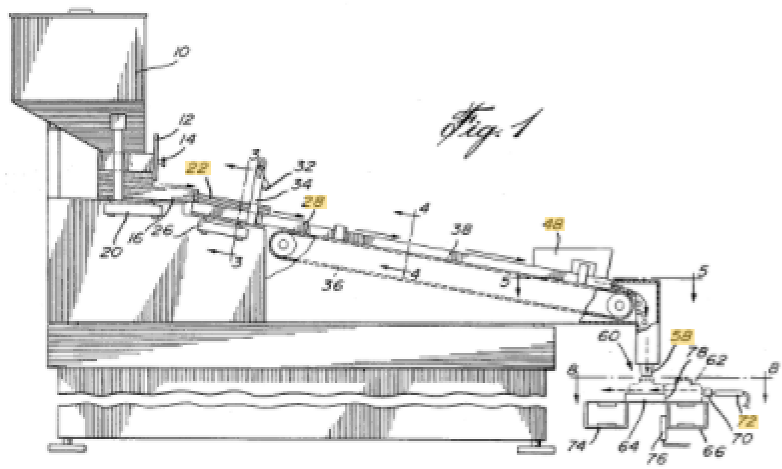


Figure: Device for counting and loading small items into containers [7]

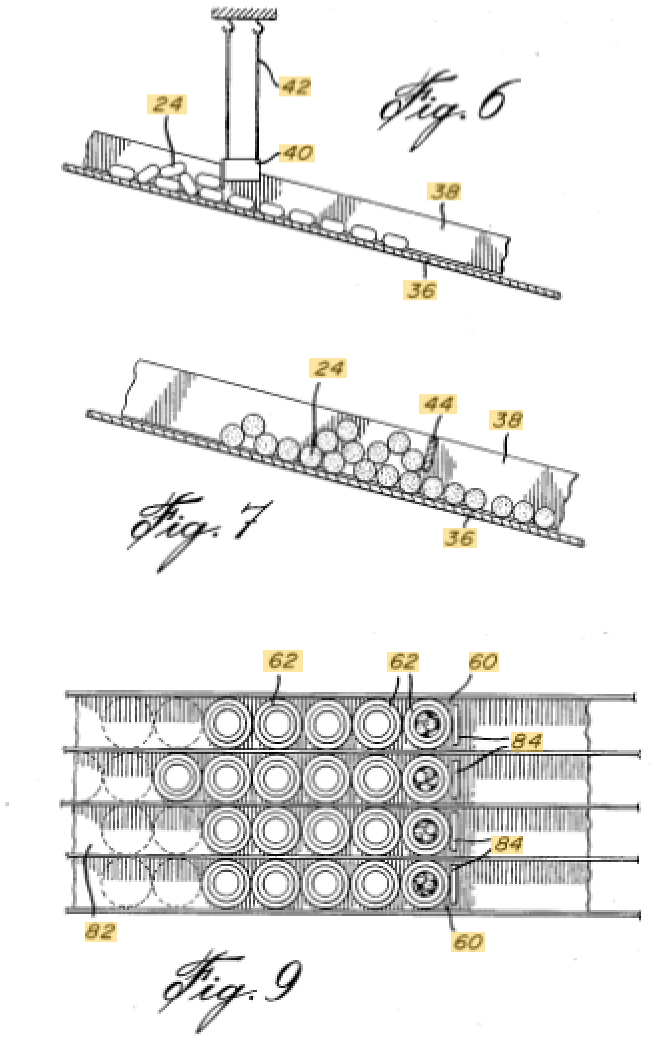


Figure: Device for counting and loading small items into containers [7]

This next device is not specifically for pills but more broadly the sorting and weighing of particulate matter. In this device a picker head grabs individual pieces of particulate matter from many small bins below the head and deposits them onto a conveyor belt for further processing. When holding the particulate the picker head obtains the weight of the particulate which allows for the particulate to be sorted by weight. This patent was granted on 2011-05-03 and is still active. Although this device is not intended for medication the mechanisms involved could easily be adapted to do so.

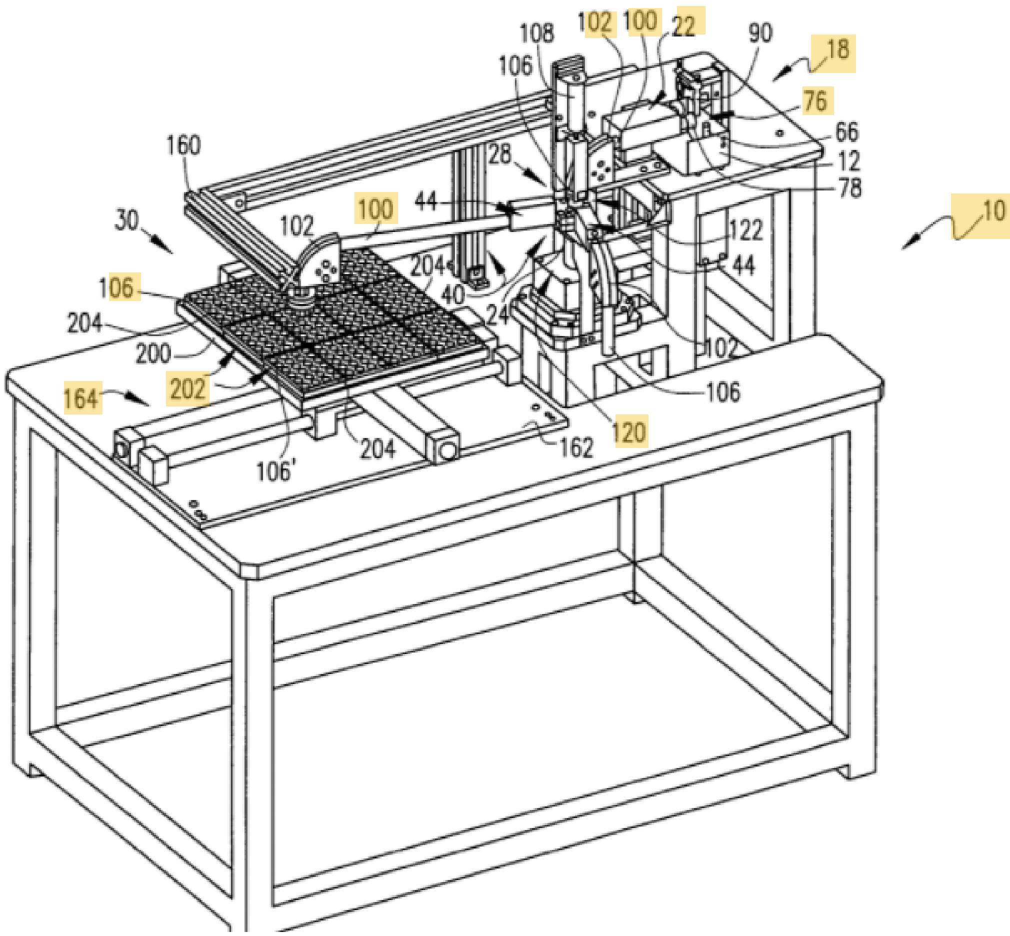


Figure: Automated picking, weighing and sorting system for particulate matter [8]

In this next device many bins of pills are configured for individual dispensing into many channels which dispense into one an array of locations. The primary purpose of this design is for many pill bottles to be filled simultaneously but this may only be done for a single type of medication at a time. This patent was granted on 1997-08-26 has since expired.

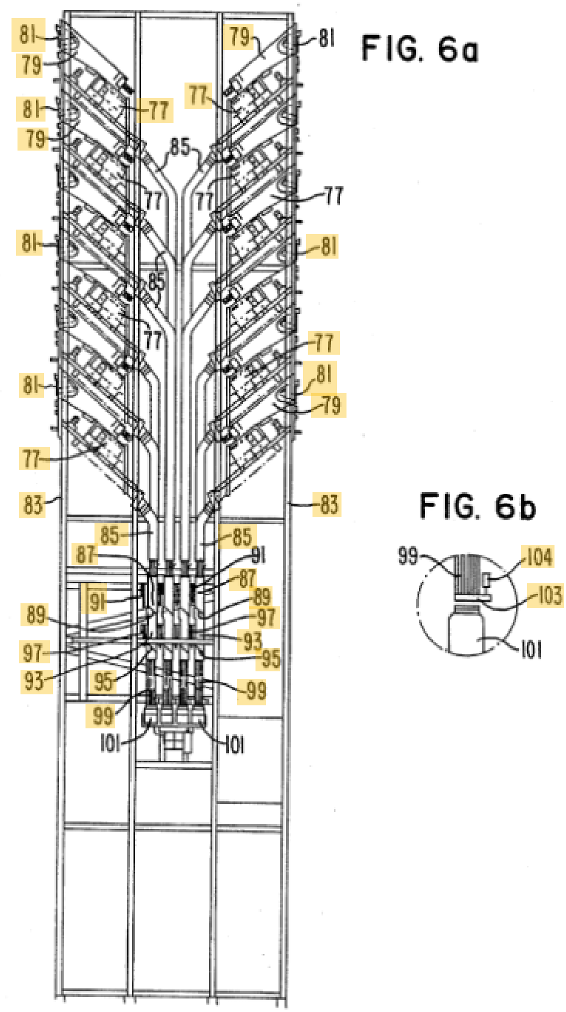


Figure: Automatic prescription dispensing system [9]

One final design to examine is this device. In this design a new pill storage/dispensing system is proposed that is the shape of a cassette tape. The system accepts only a singular type of pill through a large throat tube to fill the internal reservoir. When a pill is needed to be dispensed a internal pill conveying wheel is rotated allowing for only one pill to be dispensed at a time. This patent was granted on 1987-10-06 and has since expired.

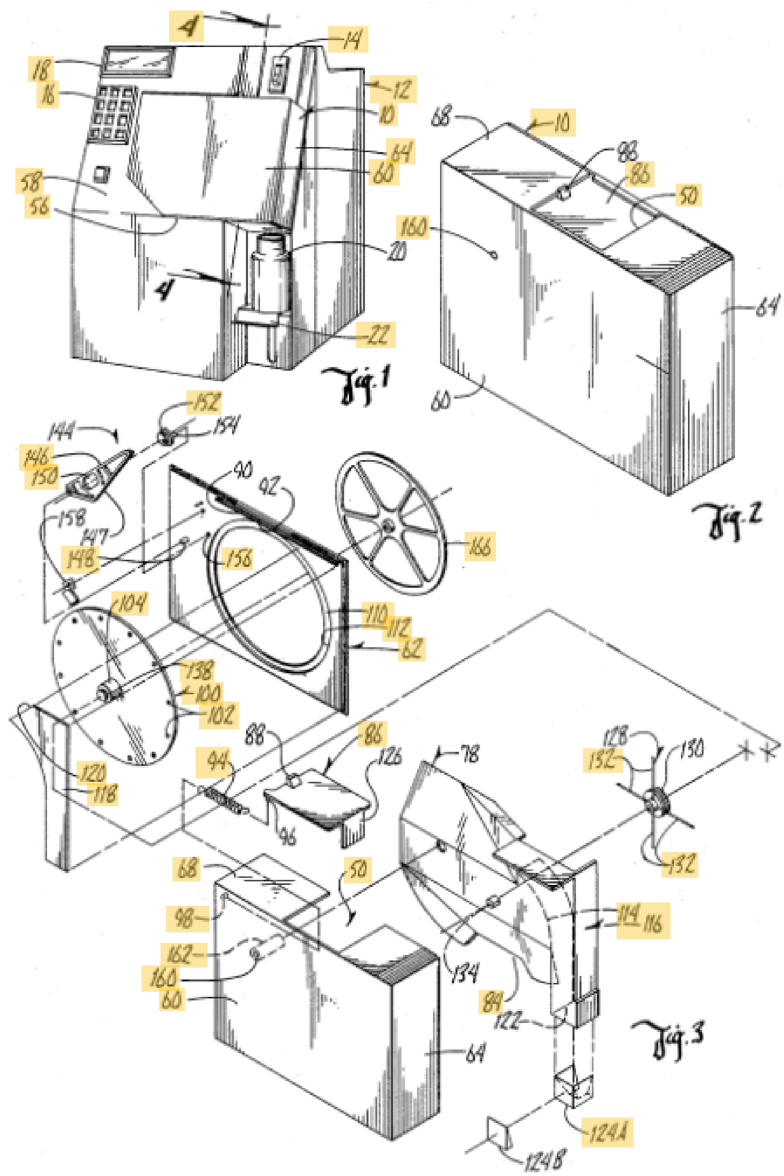


Figure: Pill storage and dispensing cassette [10]

Patent References:

[1]. https://patents.google.com/patent/USRE42766E1/en?q=pill+sorting&oq=pill+sorting
[2]. https://patents.google.com/patent/US10636519B2/en?q=pill+sorting&oq=pill+sorting
[3]. https://patents.google.com/patent/KR100844447B1/en?q=pill+sorting&oq=pill+sorting
[4]. https://patents.google.com/patent/US8091733B2/en?q=pill+sorting&oq=pill+sorting&page=1
[5]. https://patents.google.com/patent/US20120159907A1/en?q=pill+sorting&oq=pill+sorting&page=1
[6]. https://patents.google.com/patent/EP1389583B1/en?q=automatic+pill+sorting&oq=automatic+pill+sorting
[7]. https://patents.google.com/patent/US4677283A/en?q=automatic+pill+sorting&oq=automatic+pill+sorting&page=1
[8]. https://patents.google.com/patent/US7934600B2/en?q=automatic+pill+sorting&oq=automatic+pill+sorting&page=1
[9]. https://patents.google.com/patent/US5660305A/en?q=automatic+pill+sorting&oq=automatic+pill+sorting&page=1
[10]. https://patents.google.com/patent/US4697721A/en?q=automatic+pill+sorting&oq=automatic+pill+sorting&page=2

Appendix 7 - Ideas Generated

One possible approach to the sorting process would take inspiration from a childhood icon. PEZ dispensers were designed to individually dispense tiny candies each time a lever was activated. Following this method, pills can be stored in a spring loaded cylinder, with an actuator prepared to distribute set quantities.

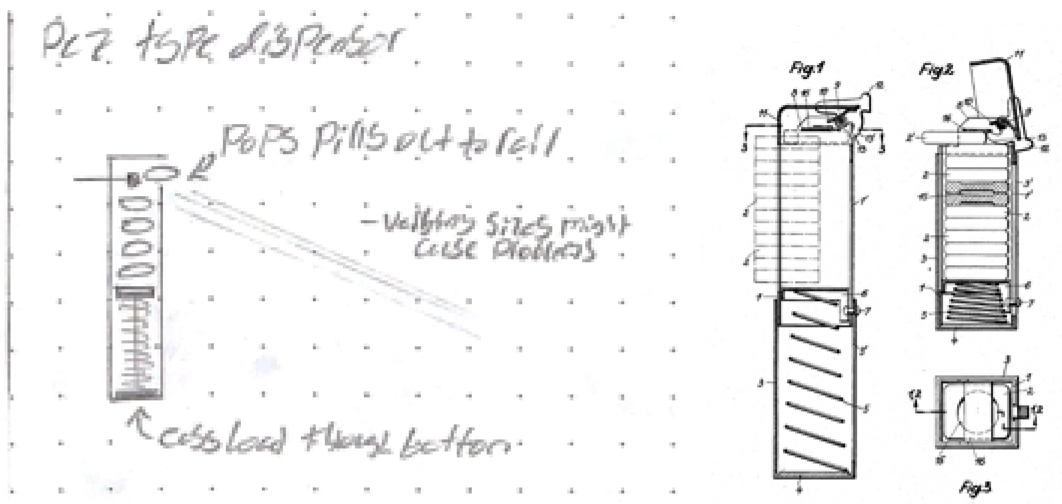


Figure: Idea #1

The pick & place type machine would allow for rapid swapping between available pills. Depending on the storage method, this design could be approached with either two or three axises in mind. In preparation for sorting, each pill type could be stored together in dishes or individual housings. The style allows for easy expansion to accommodate larger quantities. The larger work surface would also provide ample space to integrate the necessary sensors to provide redundancies.

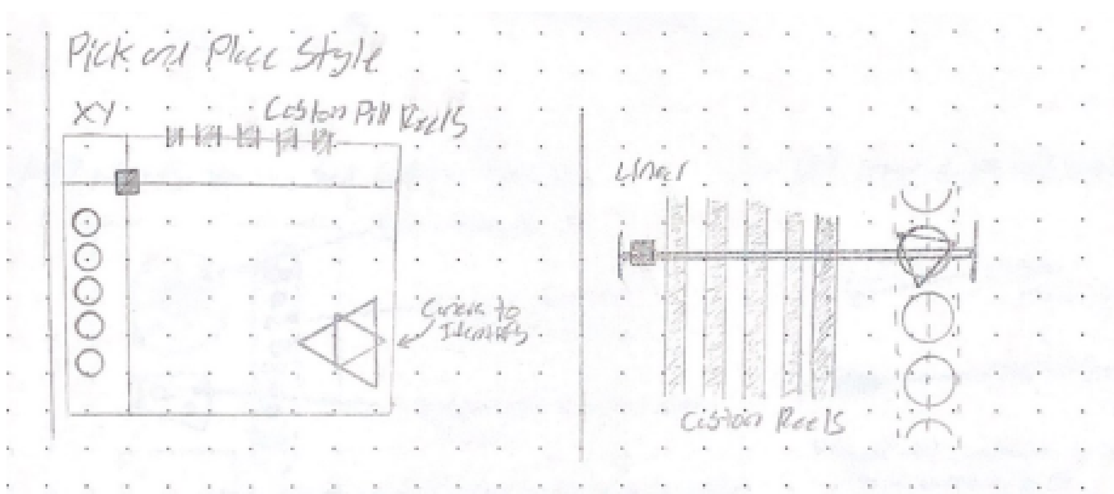


Figure: Idea #2

The final method mimics a coin pusher arcade game. Pills are sorted in preassigned slots with actuators, similar to the first concept, prepared to raise the first one in the line. When doses are requested, the actuators raise each pill simultaneously, giving the system a chance to double check everything is right. Once the quantities have been confirmed, a long arm sweeps all of the designated pills into a chute.

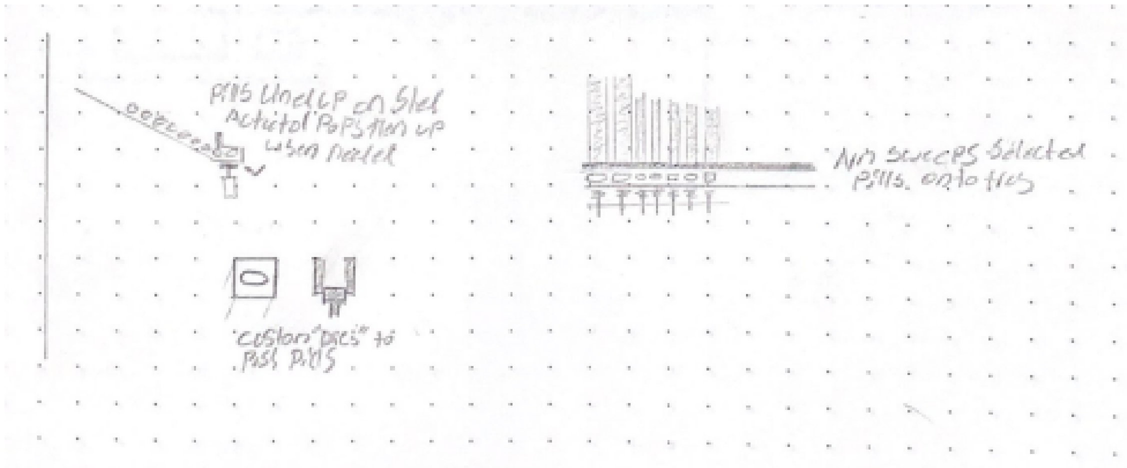
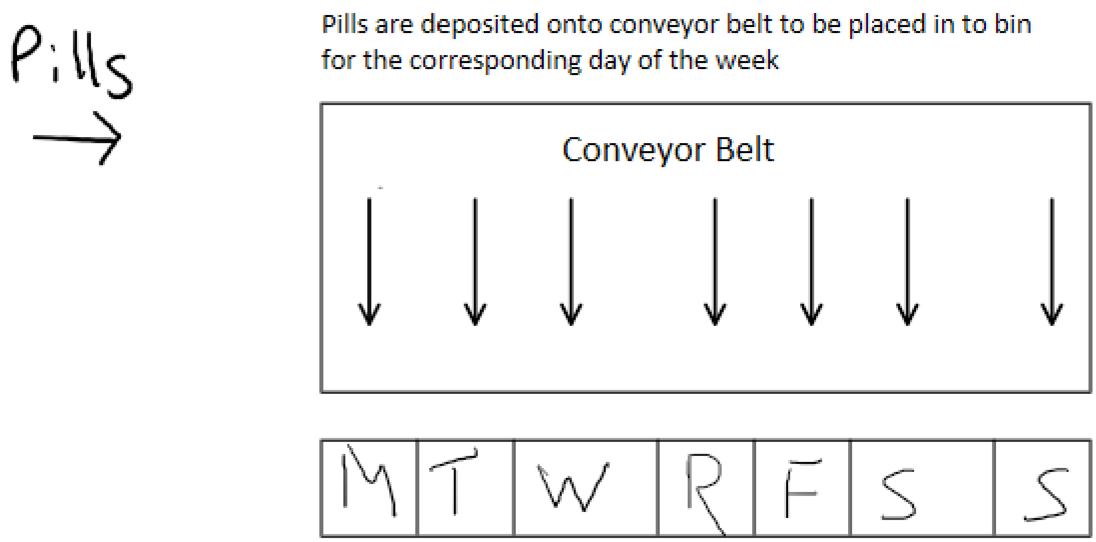
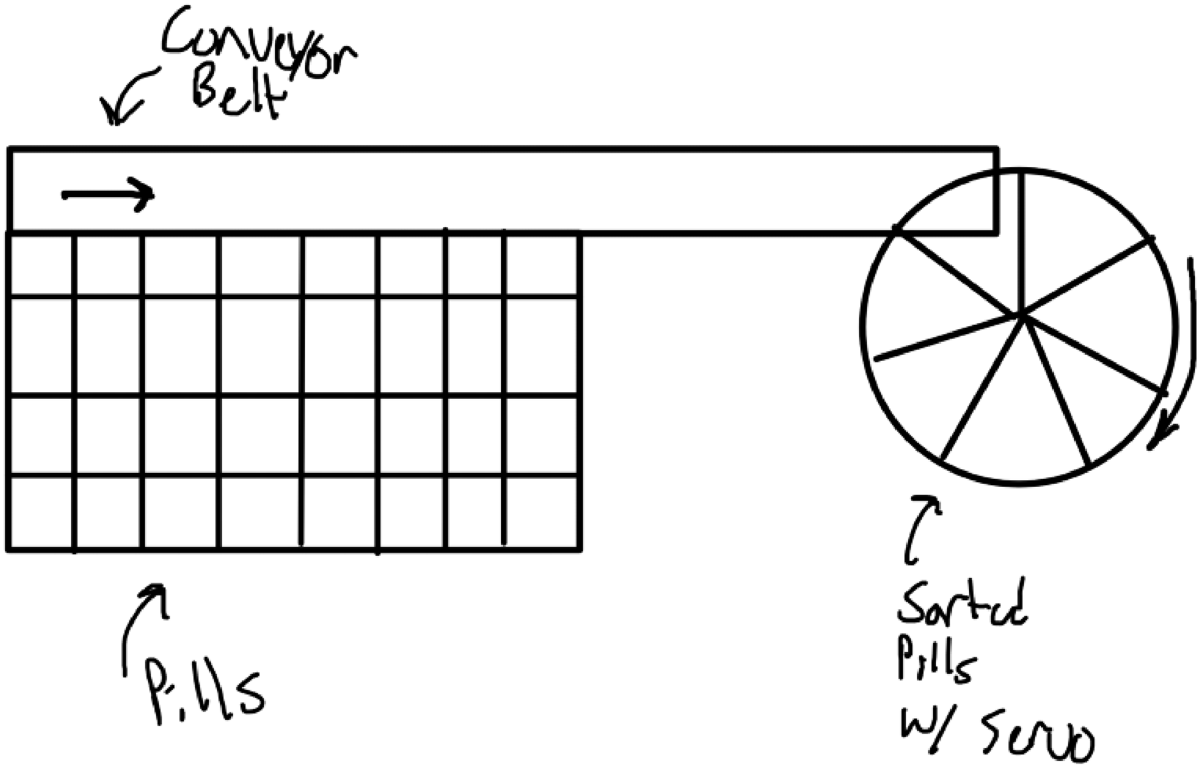
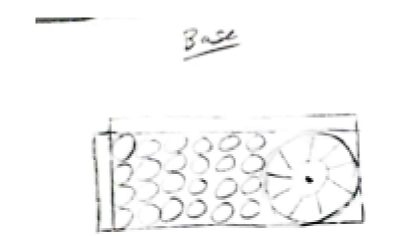
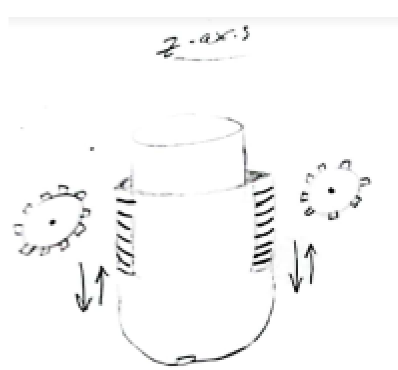


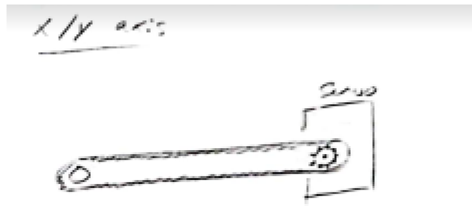
Figure: Idea #3

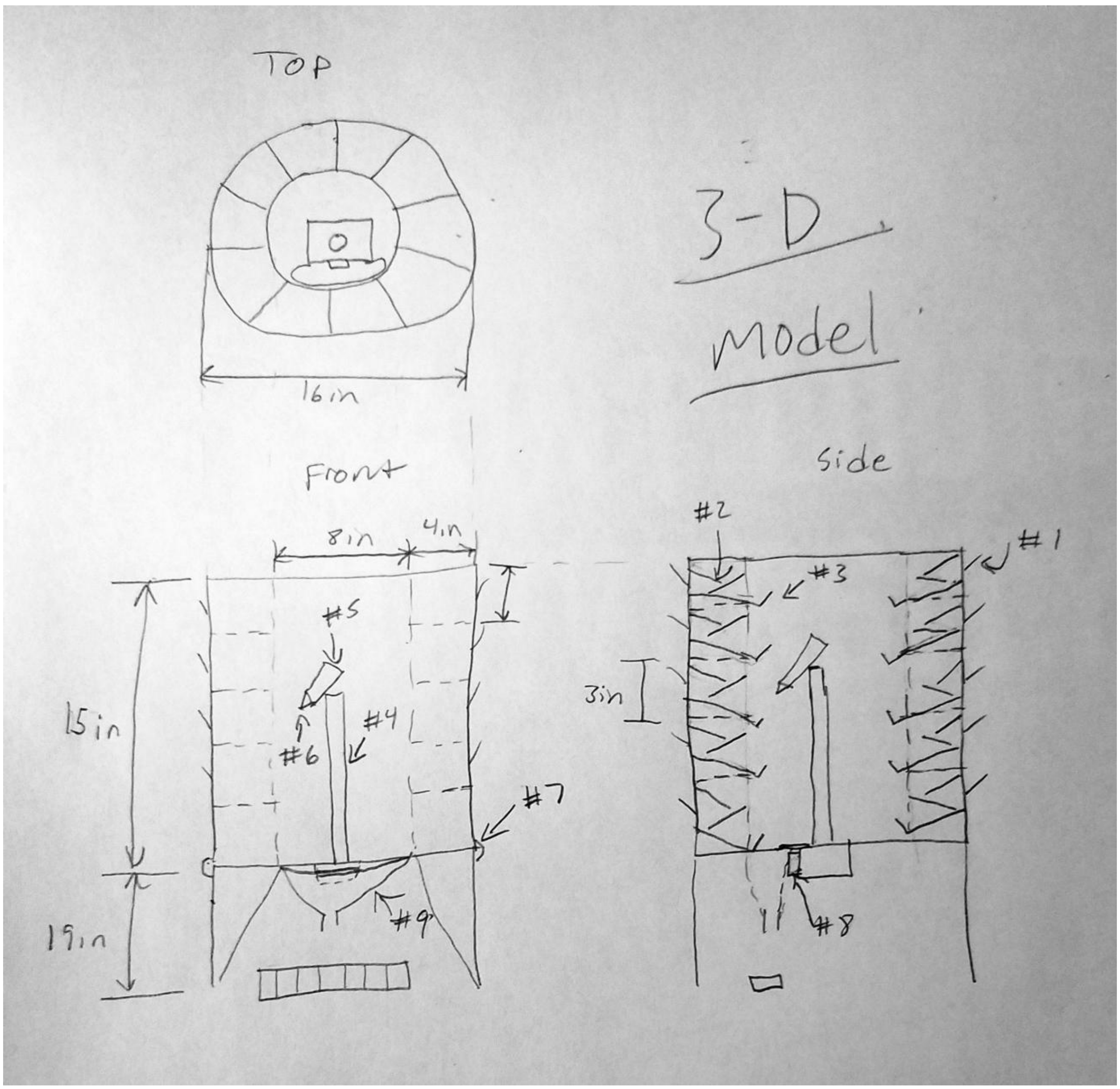



Conveyor Belt
Pills
Sorted Pills W/ Servo


Base

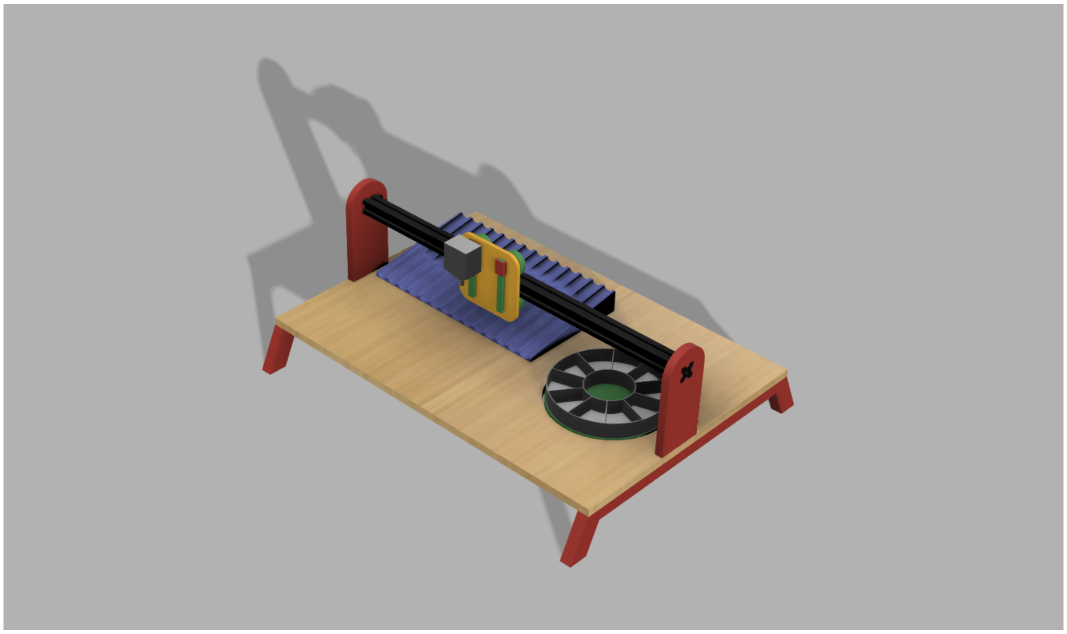

Z-axis

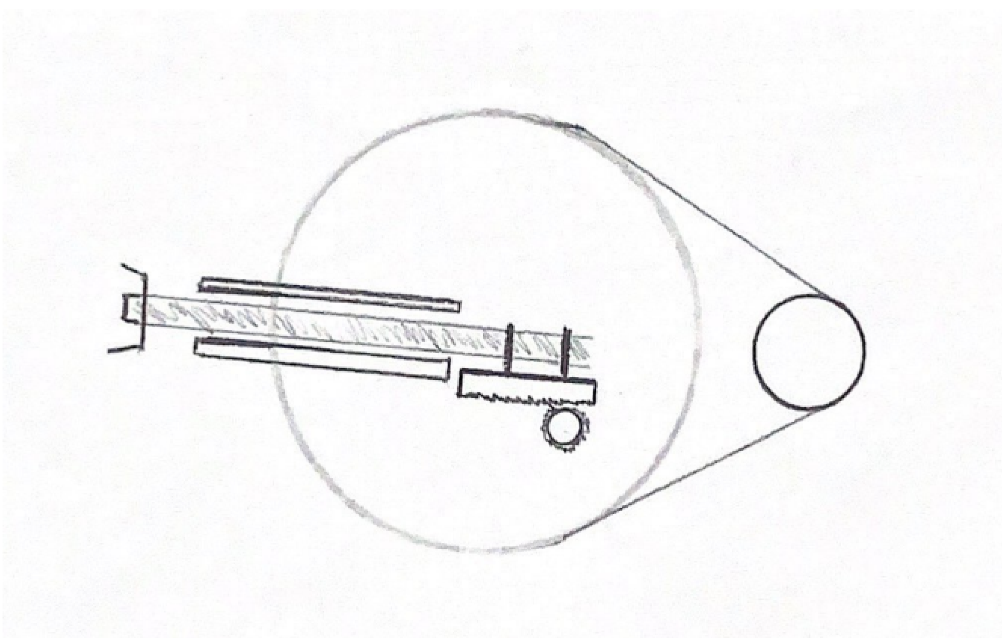


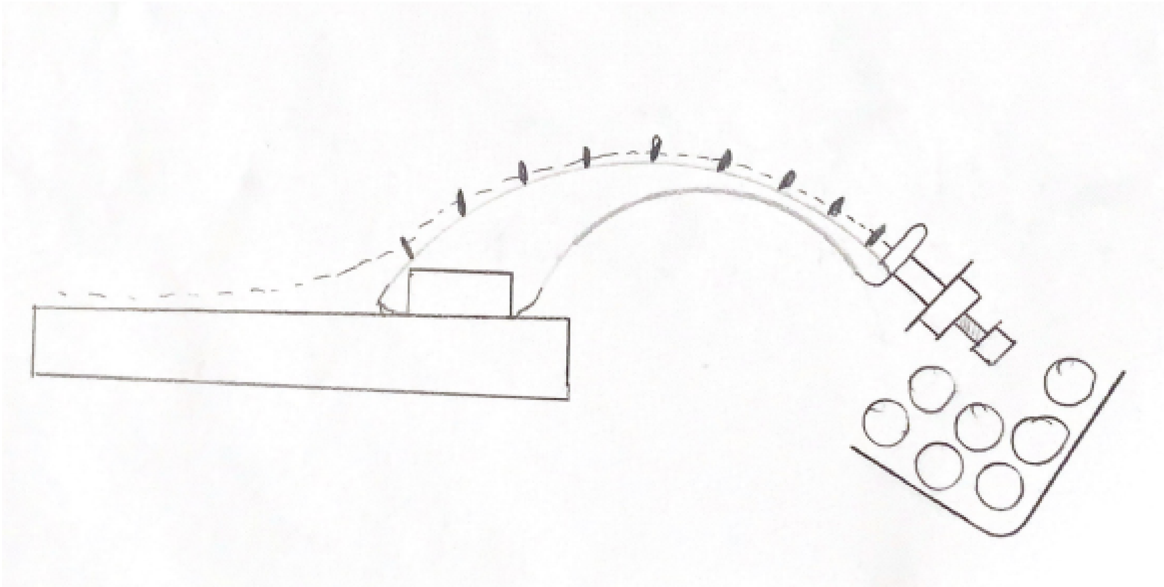


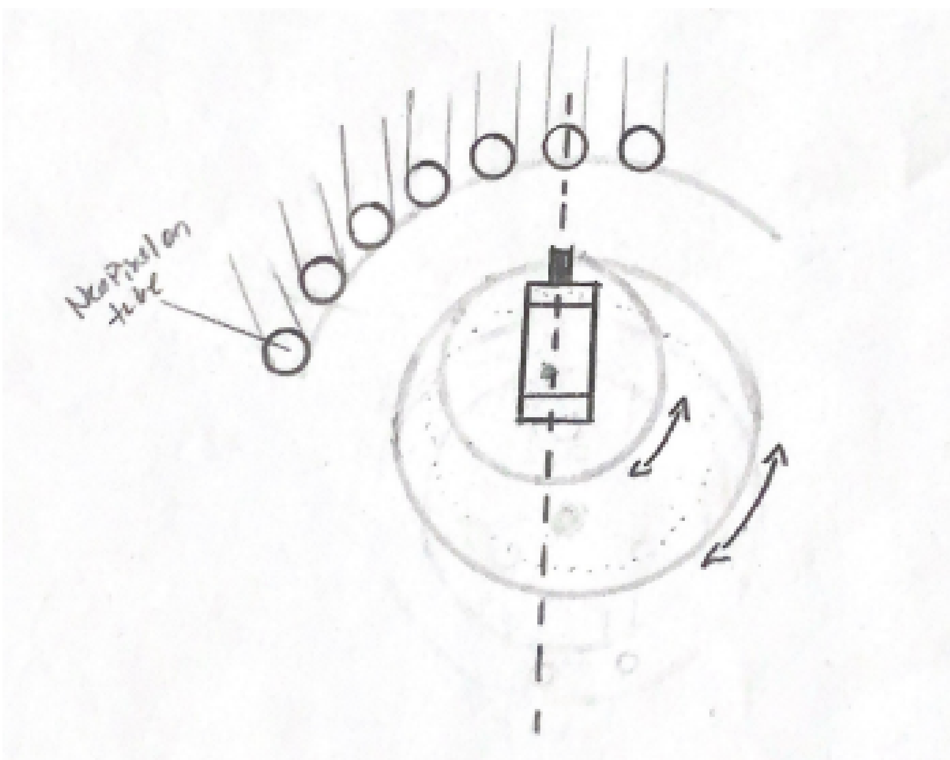
 # 1 = filling the bins with pills.
 # 2 = In the bins, there is the slide where pills are sliding to the stopper.
 # 3 = the stopper that stops the pills from falling.
 # 4 = the rod extends up and down.
 # 5 = the grabber that moves forward and backward.

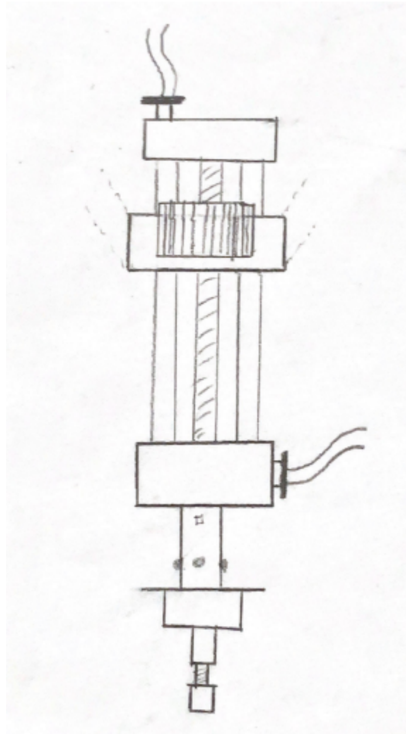
 # 6 = part of the grabber that picks up and drops the pill.
 # 7 = the cylinder that rotates right and left.
 # 8 = a motor in which the funnel is attached and moves left to right.
 # 9 = the funnel that catches the pill that is dropped from the grabber and redirects the pills to the tray.








NeoPixel on
tube



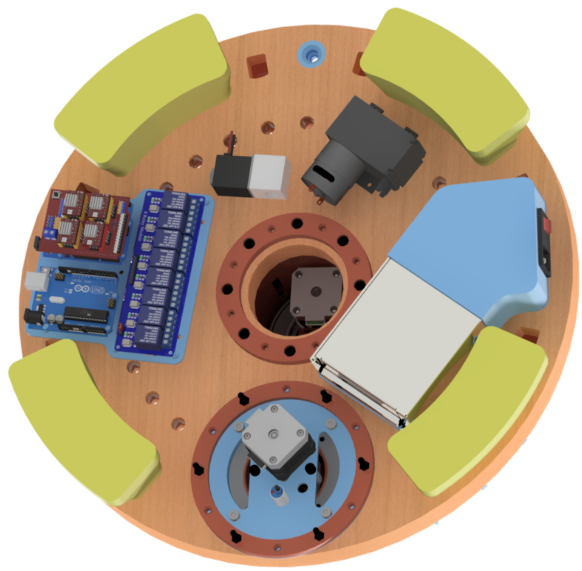
References

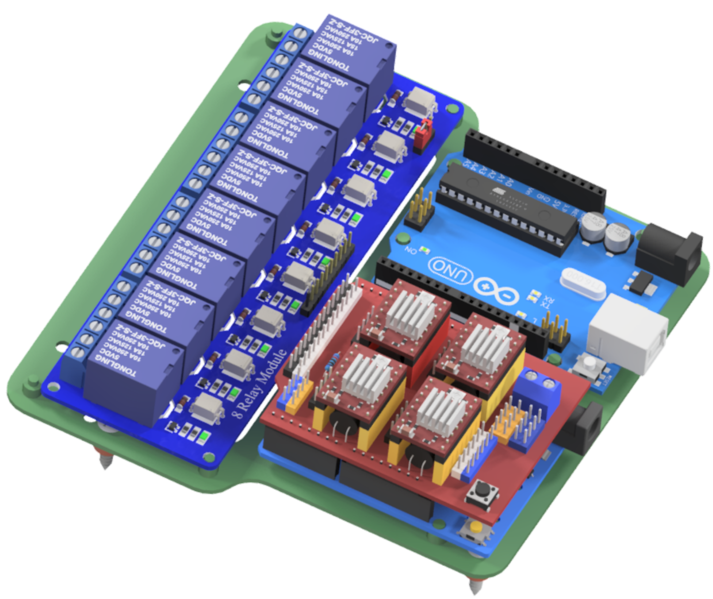
[1]: https://www.pexels.com/photo/gummy-bears-on-a-pill-organizer-7723337/

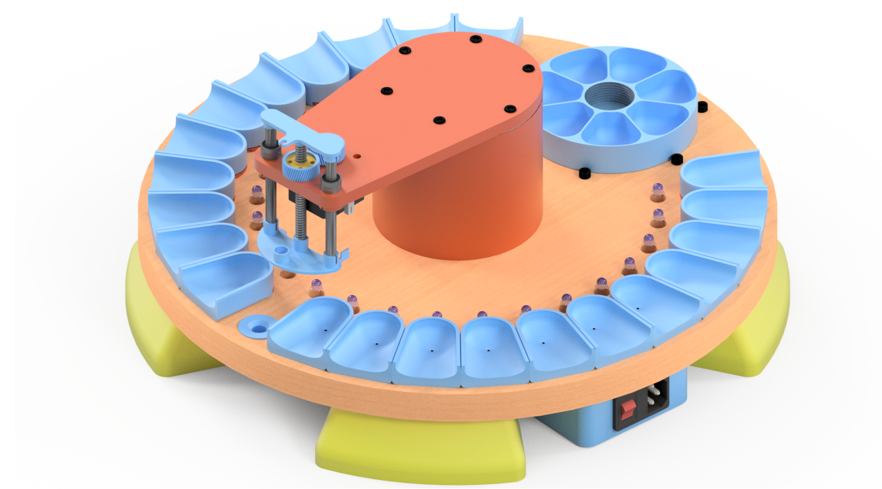
[2]: https://www.lfacapsulefillers.com/capsule-size-chart

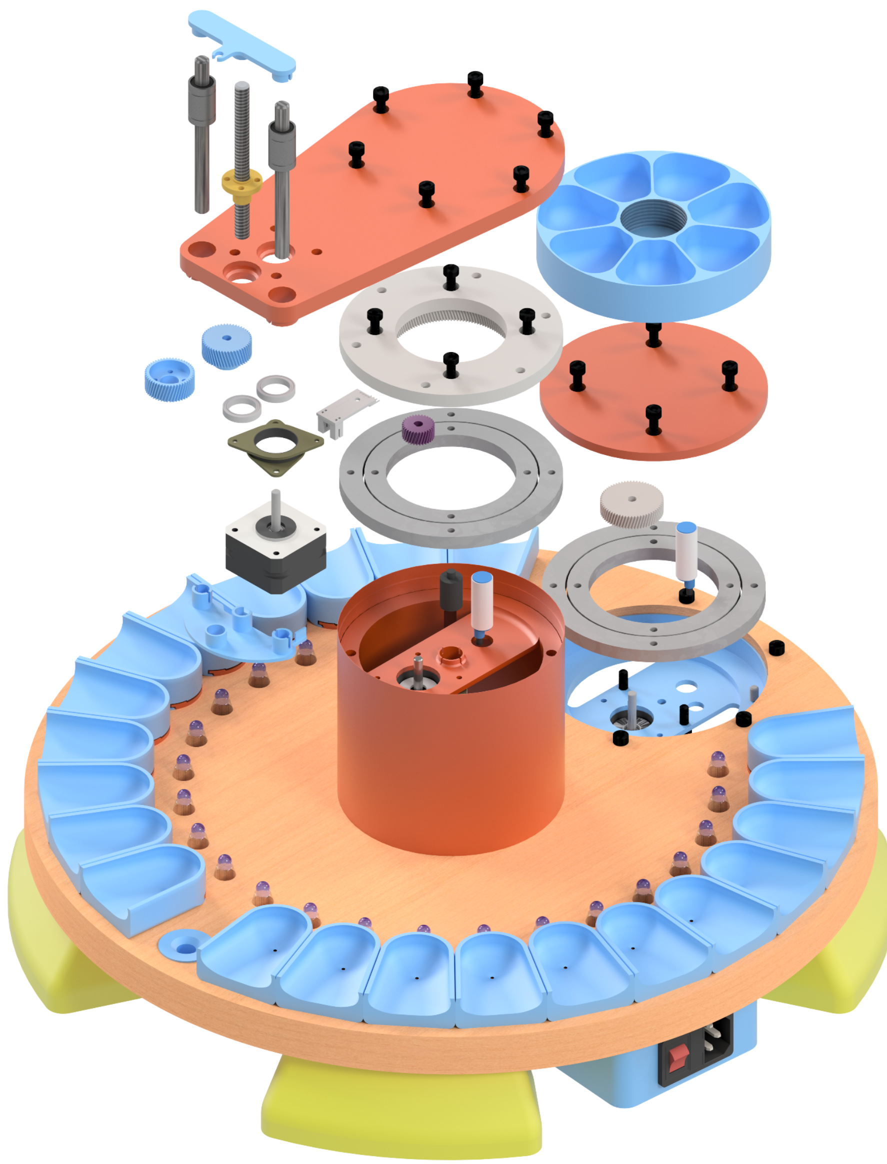
[3]: https://www.researchgate.net/figure/Figure1-Tablets-of-different-sizes-From-left-to-right-2mm-mini-tablet-4mm-mini-tablet_fig2_298425792#:~:text=of%20different%20sizes.-,From%20left%20to%20right%3A%202%20mm%20mini%2Dtablet%3B%204,tablet%20(paracetamol%20500%20mg).

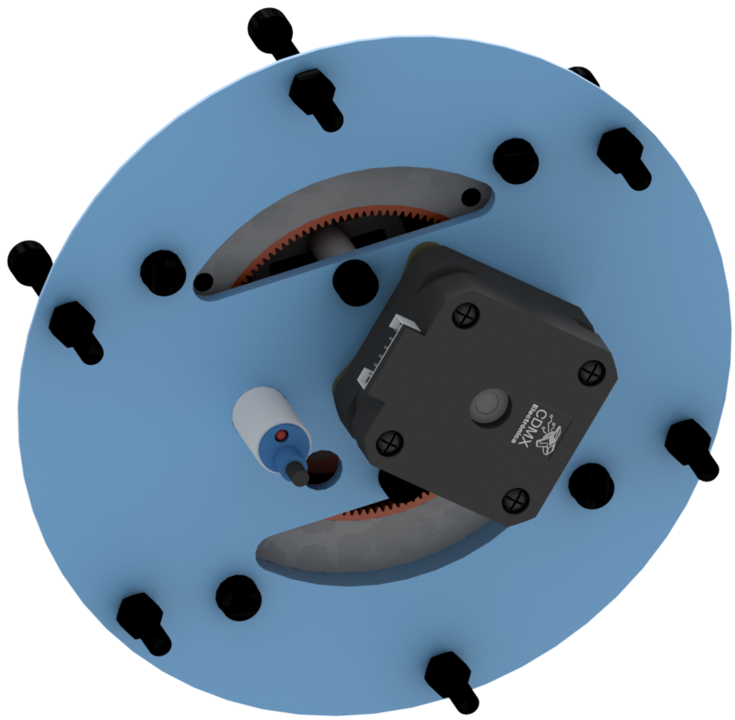
Appendix 8 - Rendered Images

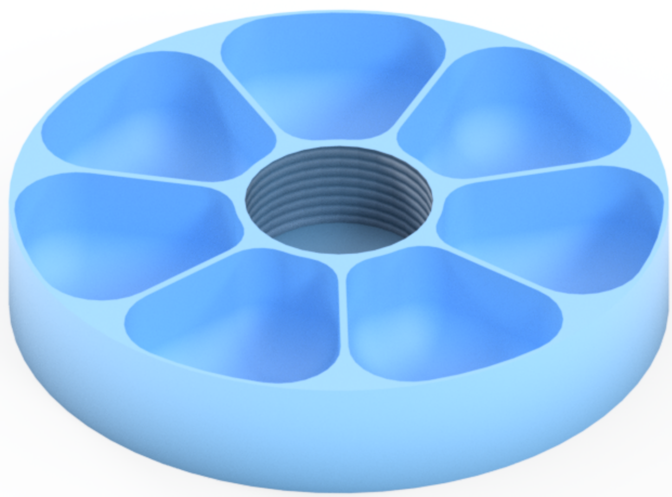


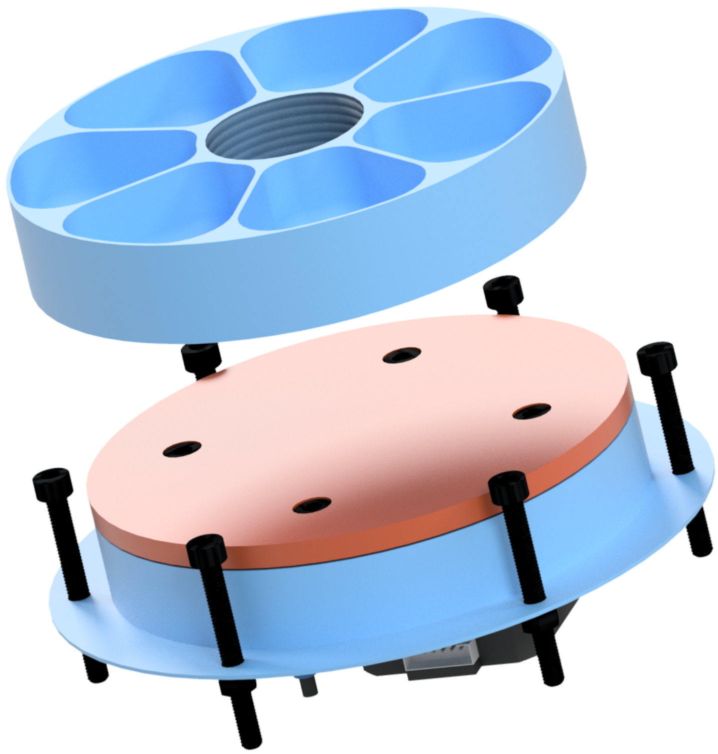


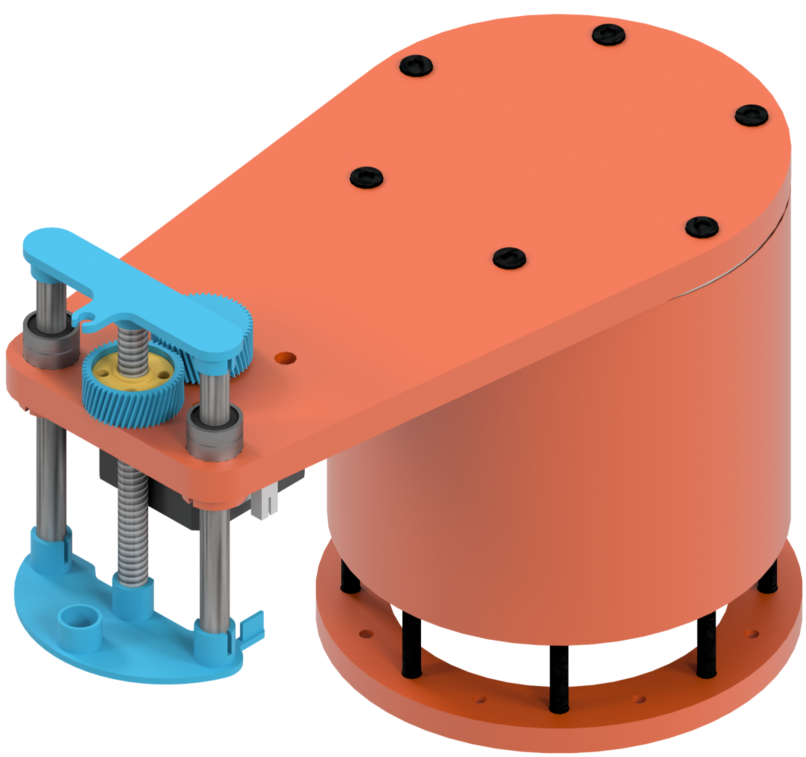






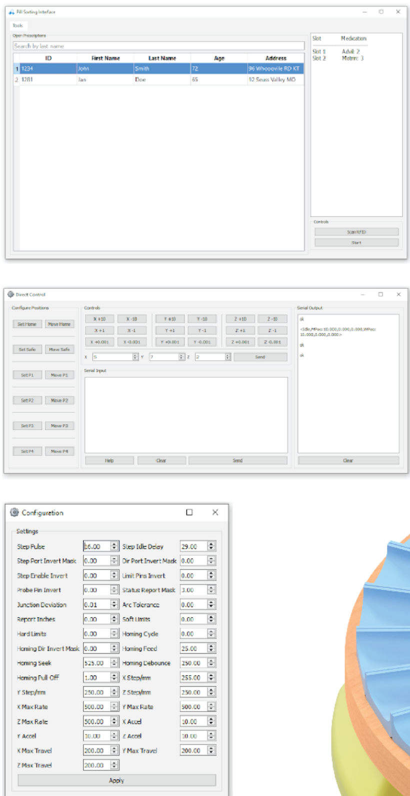


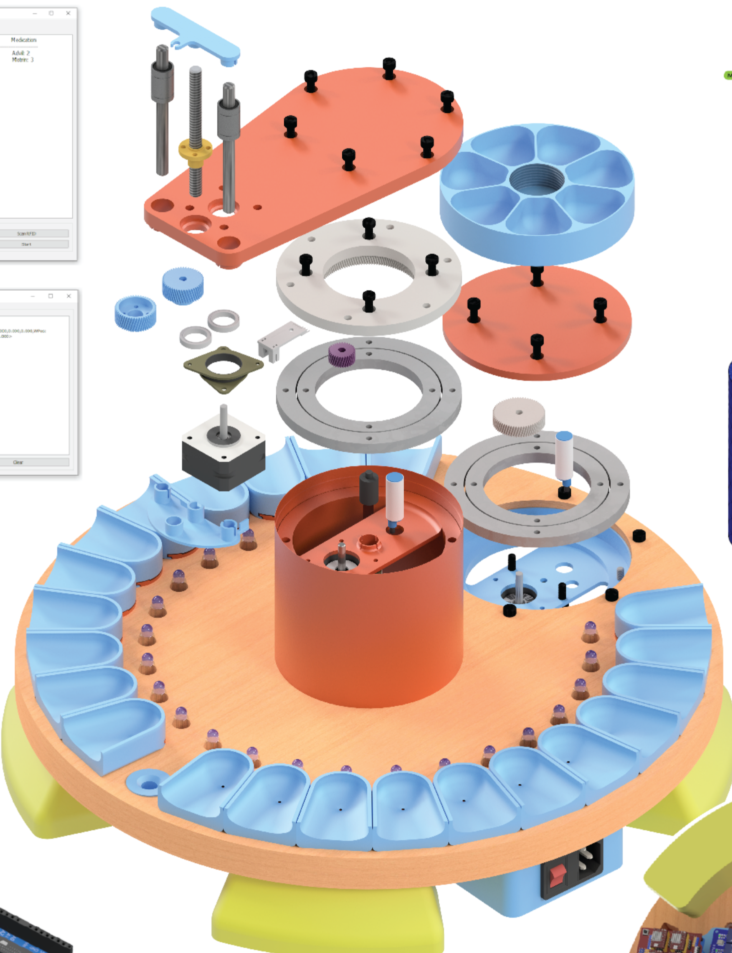


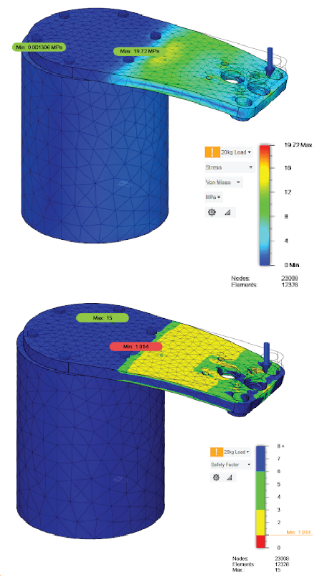


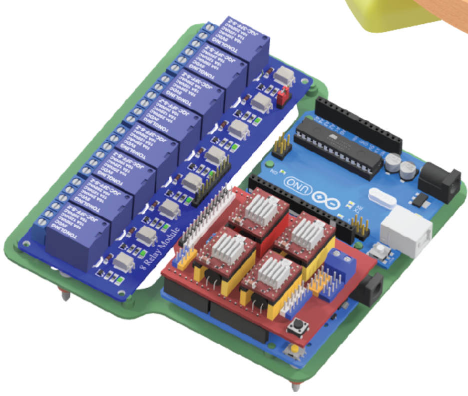
Automated Pill Sorting

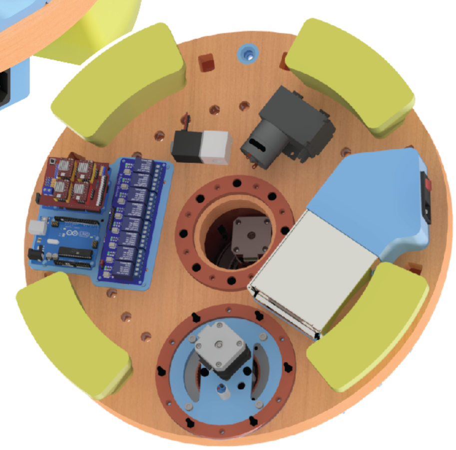
Robert McCormick • Timothy Metzger • Akash Sookun











SAINT LOUIS UNIVERSITY.
PARKS COLLEGE OF ENGINEERING, AVIATION AND TECHNOLOGY